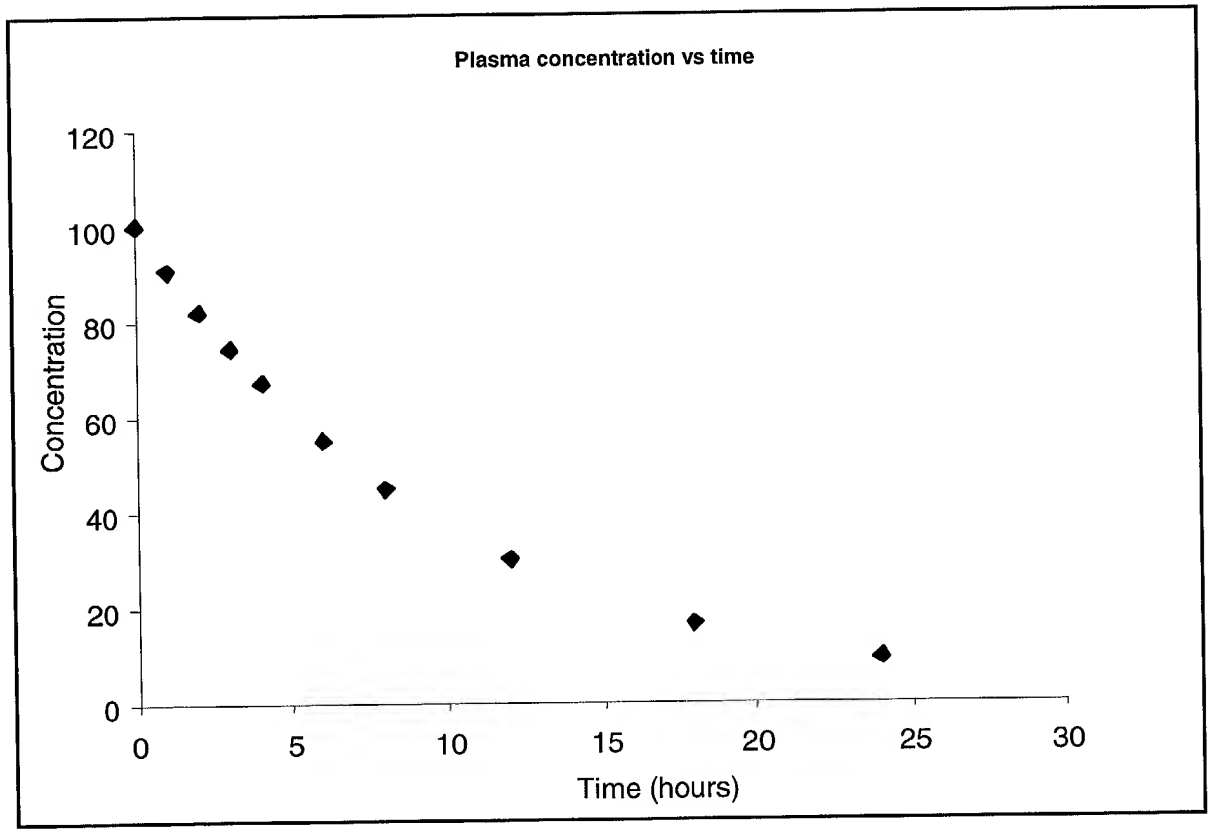

Figure 1

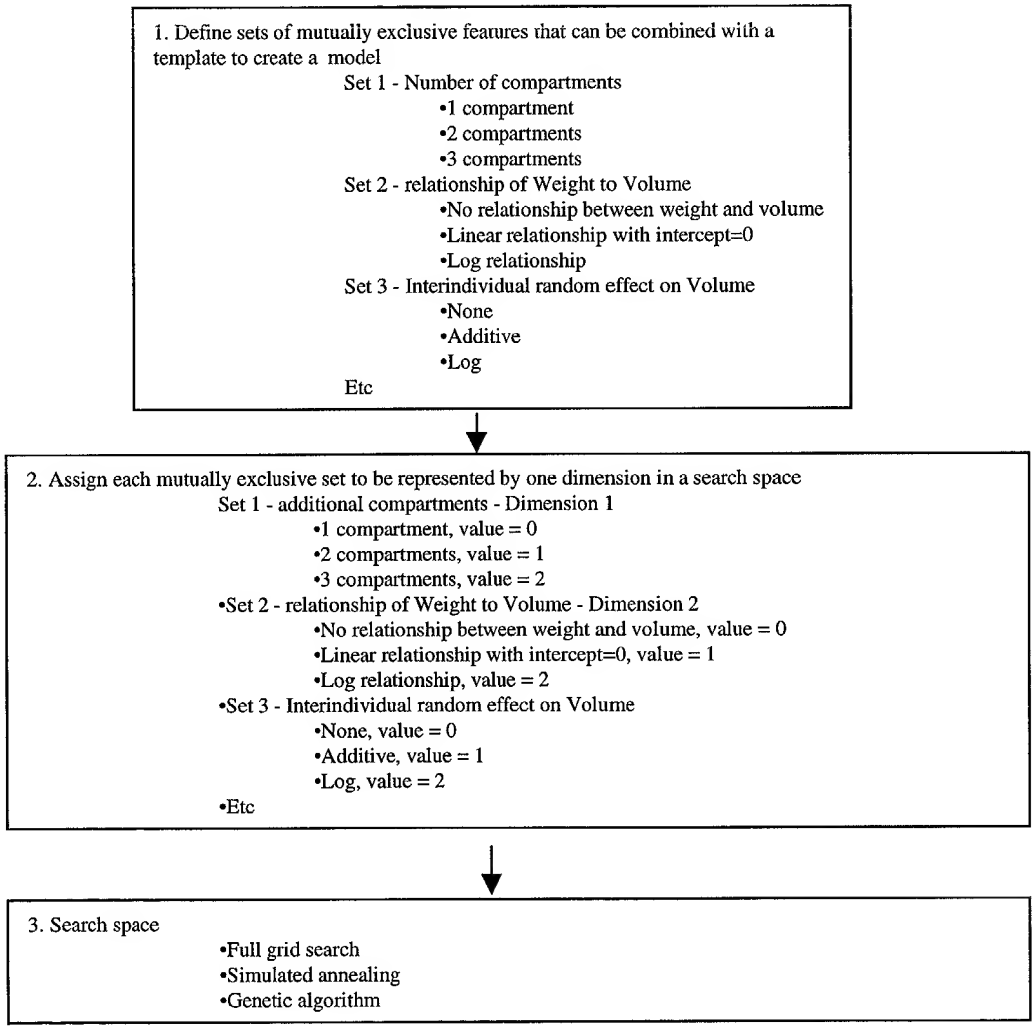
1. Define sets of mutually exclusive features that can be combined with a template to create a model
Set 1 - Number of compartments
•1 compartment
•2 compartments
•3 compartments
Set 2 - relationship of Weight to Volume
•No relationship between weight and volume
•Linear relationship with intercept=0
•Log relationship
Set 3 - Interindividual random effect on Volume
•None
•Additive
•Log
Etc
2. Assign each mutually exclusive set to be represented by one dimension in a search space
Set 1 - additional compartments - Dimension 1
•1 compartment, value = 0
•2 compartments, value = 1
•3 compartments, value = 2
•Set 2 - relationship of Weight to Volume - Dimension 2
•No relationship between weight and volume, value = 0
•Linear relationship with intercept=0, value = 1
•Log relationship, value = 2
•Set 3 - Interindividual random effect on Volume
•None, value = 0
•Additive, value = 1
•Log, value = 2
•Etc
3. Search space
•Full grid search
•Simulated annealing
•Genetic algorithm

Figure 2

Candidate model search space

3 dimensions

- # of additional compartments
- Relationship between weight and volume
- Interindividual random effect on volume

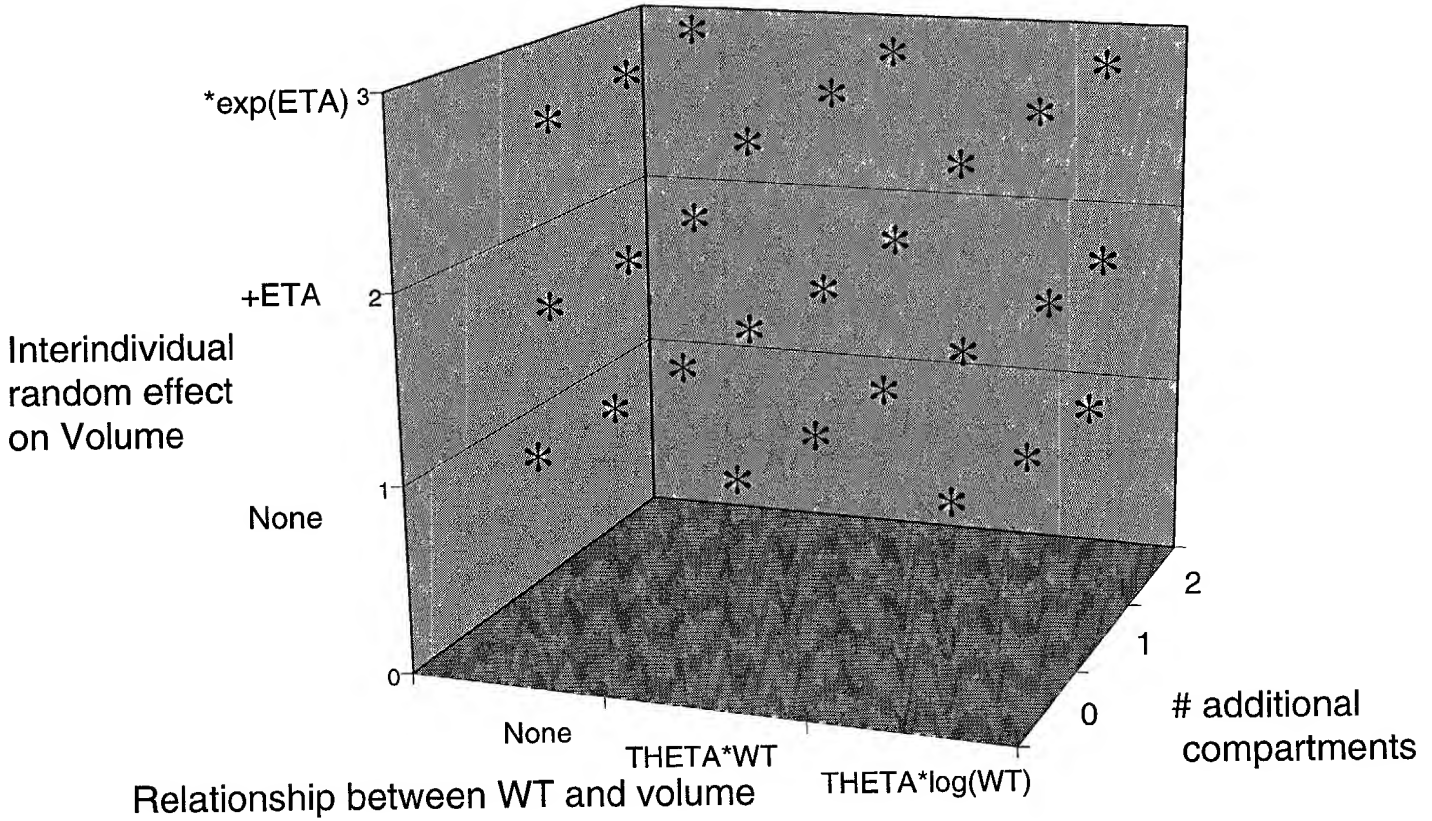

Figure 3

Loop over the values in each dimension to examine all possible combinations of values. If there are two values (0,1) in each of three dimensions, there will be 8 possible combinations

Dimension 1	Dimension 2	Dimension 3
0	0	0
0	0	1
0	1	0
0	1	1
1	0	0
1	0	1
1	1	0
1	1	1

Figure 4

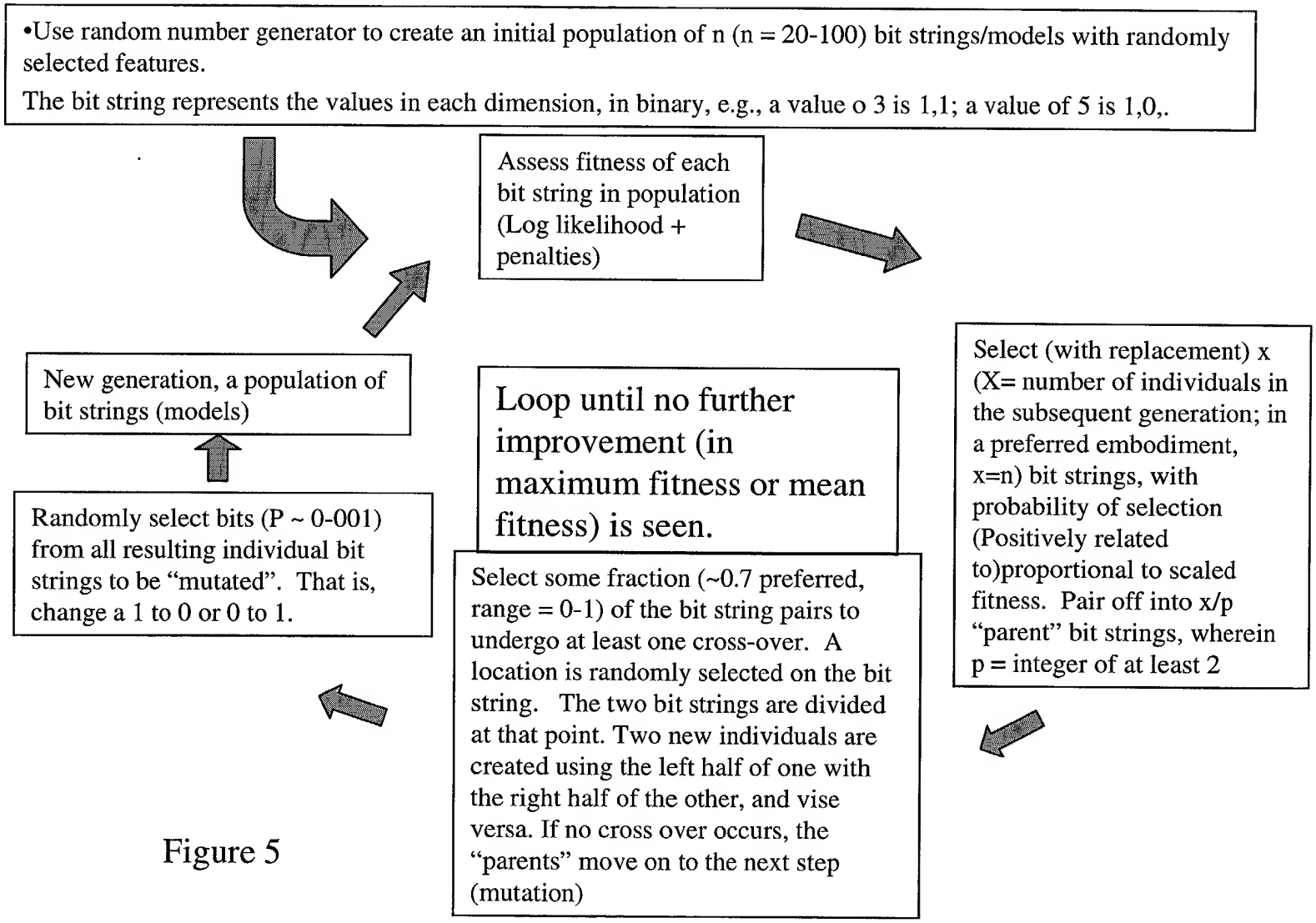

Figure 5

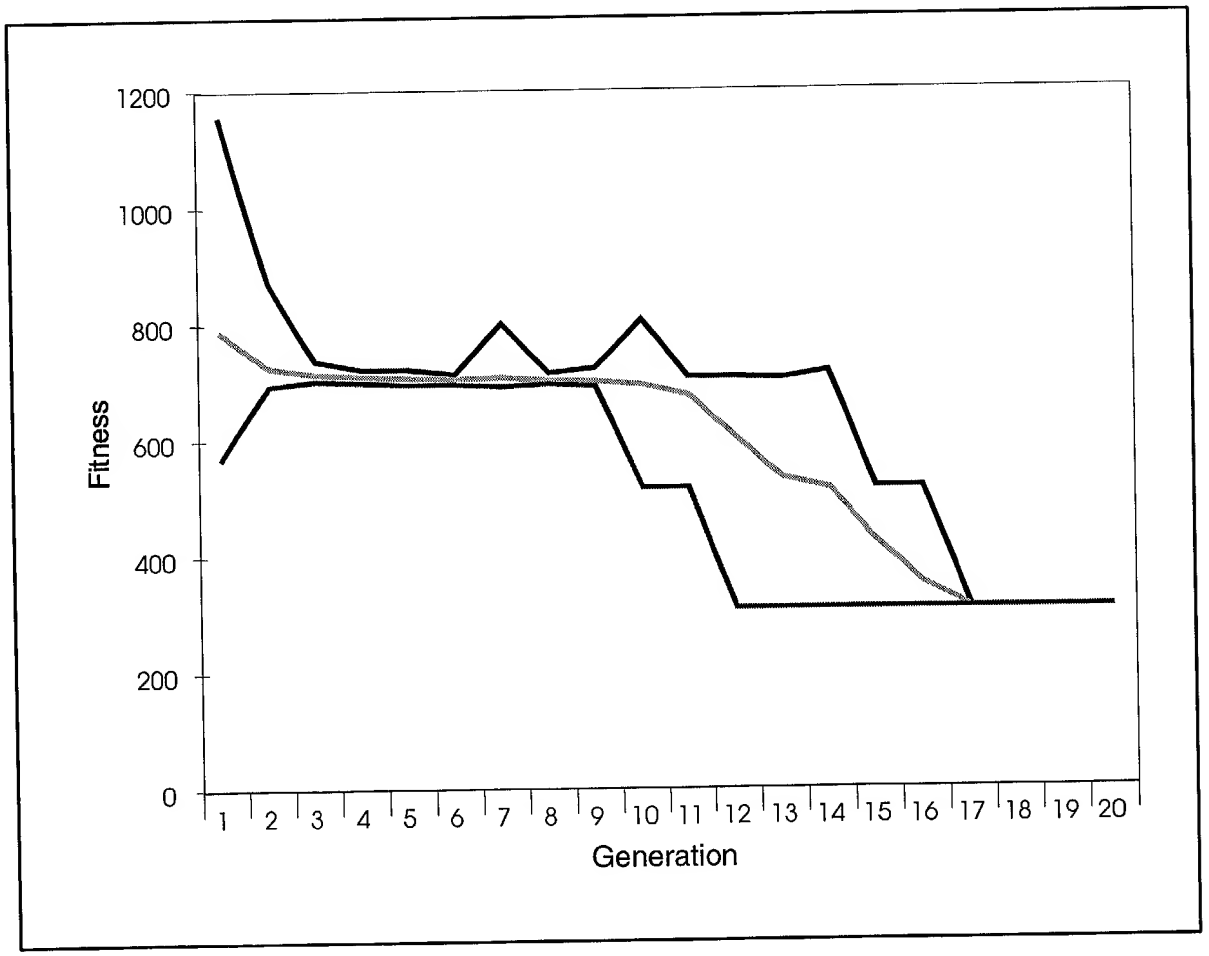

Figure 6

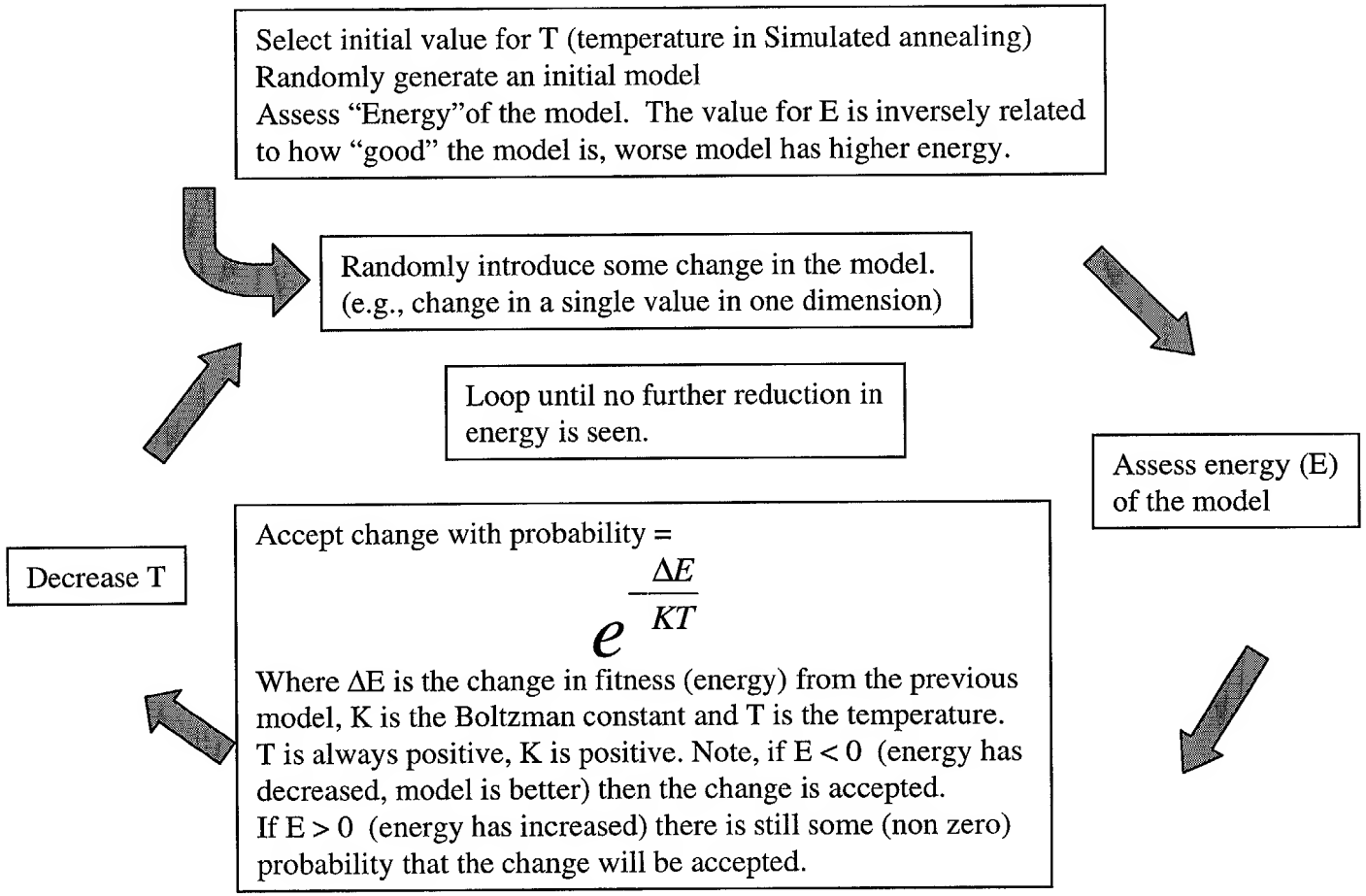

Figure 7

Figure 8

File ga_omega.vbp

```
Type=Exe
Form=frm_main.frm
Reference=*\G{00020430-0000-0000-C000-
000000000046}#2.0#0#..\WINDOWS\SYSTEM\StdOle2.tlb#OLE Automation
Module=nmga; nm_ga1.bas
Class=token_group; token_group.cls
Form=frm_tokens.frm
Form=frm_new_group.frm
Form=frm_edit_token.frm
Form=frm_options.frm
Object={B02F3647-766B-11CE-AF28-C3A2FBE76A13}#2.5#0; SS32X25.OCX
Object={02B5E320-7292-11CF-93D5-0020AF99504A}#1.0#0; MSCHART.OCX
Object={BDC217C8-ED16-11CD-956C-0000C04E4C0A}#1.1#0; TABCTL32.OCX
Object={6B7E6392-850A-101B-AFC0-4210102A8DA7}#1.2#0; COMCTL32.OCX
Form=frm_text.frm
Reference=*\G{0D452EE1-E08F-101A-852E-
02608C4D0BB4}#2.0#0#..\WINDOWS\SYSTEM\FM20.DLL#Microsoft Forms 2.0
Object Library
Form=frm_graphics.frm
Object={827E9F53-96A4-11CF-823E-000021570103}#1.0#0; GRAPHS32.OCX
Form=frm_histo.frm
Form=frm_sort_results.frm
Form=frm_debug.frm
Object={F9043C88-F6F2-101A-A3C9-08002B2F49FB}#1.1#0; Comdlg32.ocx
Object={D6EEA3C0-6216-11CF-BE62-0080C72EDD2D}#1.0#0; MARQUEE.OCX
IconForm="frm_main"
Startup="Sub Main"
HelpFile=""
ExeName32="NM_GA.exe"
Command32=""
Name="nm_ga"
HelpContextID="0"
CompatibleMode="0"
MajorVer=1
MinorVer=0
RevisionVer=0
AutoIncrementVer=0
ServerSupportFiles=0
VersionCompanyName="MGA Software"
CompilationType=0
OptimizationType=0
FavorPentiumPro(tm)=0
CodeViewDebugInfo=0
NoAliasing=0
BoundsCheck=0
OverflowCheck=0
FlPointCheck=0
FDIVCheck=0
UnroundedFP=0
StartMode=0
Unattended=0
ThreadPerObject=0
```

```
Public seed_type As String
Public seed_value As Integer
Public save_best As Boolean


Sub Main()
Dim this_file As Integer
n_files = GetSetting(appname:="NM_GA", section:="Startup", _
                Key:="N", Default:=0)
For this_file = 1 To n_files
start_files(this_file) = GetSetting(appname:="NM_GA", section:="Startup", _
          Key:="File" & str(this_file), Default:="")
frm_main.files(this_file).Visible = True
frm_main.files(this_file).Caption = start_files(this_file)
Next this_file
 home_directory = "c:\"

frm_main.Show
End Sub

Sub set_options()
With frm_options
.txt_mutation_rate = mutation_rate
.txt_cross_over_freq = cross_over_freq
.txt_frame_shift_prob = frame_shift_prob
.txt_theta_crit = theta_crit
.txt_omega_crit = omega_crit
.txt_sigma_crit = sigma_crit
.txt_cov_crit = cov_crit
.txt_generations = generation_limit
.txt_upper_limit = upper_fitness_limit
.txt_lower_limit = lower_fitness_limit
.txt_corr_crit = corr_crit
.txt_succ_crit = success_crit
If omega_block = False Then
 .chk_non_diag_omega = 0
Else
.chk_non_diag_omega = 1
End If
If save_control = False Then
.chk_save_control = 0
Else
.chk_save_control = 1
End If
If save_best = False Then
.chk_save_best = 0
Else
.chk_save_best = 1
End If
If save_output = False Then
.chk_save_output = 0
Else
.chk_save_output = 1
End If
Select Case seed_type
Case "clock"
```

```
.opt_rnd_clock = True
.txt_rnd_seed.Enabled = False
Case "user"
.opt_rnd_user = True
.txt_rnd_seed = seed_value
.txt_rnd_seed.Enabled = True
Case "default"
.opt_rnd_default = True
.txt_rnd_seed.Enabled = False
End Select
If call_method = "dll" Then .opt_dll = True
If call_method = "exe" Then .opt_exe = True
.txt_pop_size = pop_size
If n_runs = 1 Then .opt_1run = True
If n_runs = 2 Then .opt_2runs = True
If n_runs = 4 Then .opt_4runs = True
End With
End Sub


Sub get_bin(ByVal in_num As Integer, ByRef bin_str() As Boolean)
Dim n_genes As Integer, remainder As Integer, i As Integer, base As Integer
n_genes = UBound(bin_str)
base = 2
For i = 1 To n_genes
remainder = in_num Mod base
If remainder > 0 Then
bin_str(n_genes - i + 1) = True
Else
bin_str(n_genes - i + 1) = False
End If
in_num = in_num - remainder
base = base * 2
Next i

End Sub


Sub grid_search()
this_gen = 1
this_run = 0
Dim n_pop As Double, this_group As Integer, this_set As Integer
Dim binary() As Boolean, values() As Integer, max_values() As Integer
Dim n_genes As Integer, this_gene As Integer, this_ind As Integer
Dim cur_gene As Integer
n_genes = count_omega_genes() + count_non_omega_genes()
n_pop = get_n_pop()
 If n_pop < 1000000 Then
 Dim n_str As String
    If MsgBox("There will be " & n_pop & " runs" & vbCrLf & _
        "Do you want to continue?", vbOKCancel, "Full grid search") <> vbOK Then
        Exit Sub
    End If
 Else
 If n_pop > 1000000000 Then
 n_str = Format(n_pop, "Scientific")
```

```
Else
n_str = Format(n_pop, "0,000,000,000")
End If
  MsgBox ("There are " & n_str & " runs, cannot dimension genome this large, please use GA")
  Exit Sub
End If
' otherwise, continue
ReDim values(1 To n_genes, 1 To n_pop)
ReDim binary(1 To n_genes)
ReDim max_values(1 To n_genes)
ReDim fitness(1 To n_pop)

For this_gene = 1 To n_genes
  max_values(this_gene) = token_collection(this_gene).n_token_sets
Next this_gene
' first set up the first indiivdual, all 1's
For this_gene = 1 To n_genes
 values(this_gene, 1) = 1
Next this_gene
' next creat the population
' then increment each succesive individual, if you exceed max_values, increment the next
For this_ind = 2 To n_pop
  For this_gene = 1 To n_genes
    values(this_gene, this_ind) = values(this_gene, this_ind - 1)
  Next this_gene
  values(n_genes, this_ind) = values(n_genes, this_ind) + 1
  ' check to see if this is over max
  cur_gene = n_genes
  While values(cur_gene, this_ind) > max_values(cur_gene)
    values(cur_gene, this_ind) = 1
    values(cur_gene - 1, this_ind) = values(cur_gene - 1, this_ind) + 1
    cur_gene = cur_gene - 1
  Wend
Next this_ind
gen_directory = home_directory & "\1"
Dim test As String
test = Dir(gen_directory, vbDirectory)
If test <> "1" Then
 MkDir (gen_directory)
End If
 ChDir (home_directory & "\1")
' scaled fitness is dummy
Dim scaled_fitness() As Single
ReDim scaled_fitness(1 To n_pop)
' and run population
run_population scaled_fitness(), values(), False
End Sub

Sub ga_runner(start_new_run As Boolean, check_out As Boolean)
Dim scaled_fitness() As Single
Dim n_pop As Double, this_group As Integer, this_set As Integer
Dim genes() As Integer, max_values() As Integer
Dim unmapped_values() As Integer, total_bits As Long
Dim n_bits() As Integer ' number of bits in each gene
Dim total_bit As Integer, this_bit As Integer
Dim n_omega_genes As Integer, this_gene As Integer, this_ind As Integer
```

```
Dim cur_gene As Integer
Dim n_rows As Integer, max_x As Single
' set random seed if needed
If seed_type = "clock" Then Randomize
If seed_type = "user" Then Randomize (seed_value)

' need to add genes for non-diagonal omega
n_non_omega_genes = count_non_omega_genes()
n_omega_genes = count_omega_genes()
n_genes = n_non_omega_genes + n_omega_genes
' n_pop is either the maximum number or the number selected
' we need to distiguish between genes  (on genes) and bits (on genome).
' a gene is represented by one or more bits
' n_pop is the population size, first we'll see ho
n_pop = get_n_pop()
If n_pop > pop_size Then
  n_pop = pop_size
Else
  MsgBox ("Only " & n_pop & " combinations exist, will do grid search")
  grid_search
  Exit Sub
End If
' unmapped values are the "raw" values straight from the genome
' which is basically randomly generated
' you unmap them to get the "genes", which is used to create the control file.
ReDim unmapped_values(1 To n_genes, 1 To n_pop)
ReDim genes(1 To n_genes, 1 To n_pop)
ReDim n_bits(1 To n_genes) ' how many bits in each gene , only 1 if there are 2 possibilities, 2 if 3 or 4 etc
ReDim fitness(1 To n_pop)
ReDim scaled_fitness(1 To n_pop)
ReDim max_values(1 To n_genes)
ReDim fitness(1 To n_pop)
ReDim unique_fit(1 To n_pop * generation_limit, 1 To 7)
 ' genome_integer, fitness,generation,indivdual, obj, success, covar,
 '  need success and covar to put into the table. Scaled fitness will be added later,
 '  from a new calculation.
 ' find the maximum values that the gene can have, min value = 1?
 non_omega_bits = count_non_omega_bits
 omega_bits = count_omega_bits
 total_bits = non_omega_bits + omega_bits
 For this_gene = 1 To n_non_omega_genes ' do not include omega genes
   max_values(this_gene) = token_collection(this_gene).n_token_sets
   n_bits(this_gene) = get_nbits(max_values(this_gene))
 Next this_gene
 ' next genes are block genes for omega, each is only 1 bit
 For this_gene = n_non_omega_genes + 1 To n_non_omega_genes + n_omega - 1
    max_values(this_gene) = 2
    n_bits(this_gene) = 1
 Next this_gene
 ' final genes are sequence genes for omega
 Dim this_omega As Integer: this_omega = n_omega
 For this_gene = n_non_omega_genes + n_omega To n_genes  ' do not include omega genes
    max_values(this_gene) = this_omega
    this_omega = this_omega - 1
   n_bits(this_gene) = get_nbits(max_values(this_gene))
 Next this_gene
```

```
' now that we know how many bits, we can define the size of the genome (a binary)
'if genomoe is not yet defined
frm_main.pgb_gen.max = generation_limit
  frm_main.pgb_gen.min = 0
If start_new_run = True Then
 ReDim genome(1 To total_bits, 1 To n_pop)
 ' next creat the population
 ' then increment each succesive individual, if you exceed max_values, increment the next
 For this_ind = 1 To n_pop
  For this_bit = 1 To total_bits
    genome(this_bit, this_ind) = (Rnd() > 0.5)
  Next this_bit
 Next this_ind
  frm_main.pgb_gen.value = 0
Else ' continue old run, first check to see if old genome is the right size
 this_gen = last_gen
 If UBound(genome, 1) <> total_bits Or UBound(genome, 2) <> n_pop Then
 MsgBox "Error in genome, starting new GA run"
  ReDim genome(1 To total_bits, 1 To n_pop)
 '  create the population anyway, can't use old genome
 ' then increment each succesive individual, if you exceed max_values, increment the next
 For this_ind = 1 To n_pop
  For this_bit = 1 To total_bits
    genome(this_bit, this_ind) = (Rnd() > 0.5)
  Next this_bit
 Next this_ind
  frm_main.pgb_gen.value = this_gen
End If

End If
ReDim min_fitness(1 To generation_limit)
ReDim mean_fitness(1 To generation_limit)
ReDim max_fitness(1 To generation_limit)
n_rows = generation_limit
max_x = generation_limit
If start_new_run = True Then initialize_plot n_rows
While this_gen < generation_limit And stop_run = False
   this_gen = this_gen + 1
   frm_main.pgb_gen.value = this_gen
   Dim ok As String
   gen_directory = home_directory & "\" & this_gen
    ok = Dir(gen_directory, vbDirectory)
    If Trim(ok) = Trim(str(this_gen)) Then
      ok = Dir(gen_directory & "\control", vbNormal)
      If ok <> "" Then Kill gen_directory & "\*.*"
    Else
      MkDir gen_directory
      ChDir gen_directory
    End If
 ' uncode genome, write to unmapped values
 uncode unmapped_values(), n_bits()
 ' then unmap
 unmap genes(), max_values(), unmapped_values(), n_bits()
    run_population scaled_fitness, genes, check_out
    last_gen = this_gen
 '   Dim test_str As String, i As Integer
```

```
'   test_str = "before: "
'   For i = 1 To UBound(genome, 1)
'    If genome(i, 1) = True Then
'    test_str = test_str & 1
'    Else
'     test_str = test_str & 0
'     End If
'    Next i
    get_next_gen scaled_fitness ' only need genome, then goes back to uncode and unmap
'    test_str = test_str & vbCrLf & "after: "
'    For i = 1 To UBound(genome, 1)
'     If genome(i, 1) = True Then
'     test_str = test_str & 1
'     Else
'      test_str = test_str & 0
'      End If
'     Next i
'     MsgBox test_str, , "after"
    save_model "temp" & this_run & ".mdl" ' save a temp copy after every new genome defined
  Wend
  'frm_main.Show 1, frm_main

  End Sub
  Private Sub get_next_gen(scaled_fitness)
  Dim n_genes As Integer, this_ind As Integer, i As Integer
  Dim cum_fitness() As Single 'cummulative fitness, sum = 1
  Dim pairs() As Integer
  Dim n_pop As Integer
  n_genes = UBound(genome, 1)
  n_pop = UBound(scaled_fitness)
  ReDim cum_fitness(1 To n_pop)
  'Dim new_genome() As Boolean
  'ReDim new_genome(1 To n_genes, 1 To n_pop)
  ReDim pairs(1 To 2, 1 To n_pop / 2)

  If save_best = True Then
   Dim saved_genome() As Boolean: ReDim saved_genome(1 To n_genes)
   Dim max_fitness As Single, best_one As Integer
   max_fitness = -9999999
   For i = 1 To n_pop
    If scaled_fitness(i) > max_fitness Then
      max_fitness = scaled_fitness(i)
      best_one = i
    End If
   Next i
   For i = 1 To n_genes
    saved_genome(i) = genome(i, best_one)
   Next i
  End If
  'calculate cumulative fitness, scaled to 1.
  cum_fitness(1) = scaled_fitness(1)
  For i = 2 To n_pop
  cum_fitness(i) = cum_fitness(i - 1) + scaled_fitness(i)
  Next i
  'and divide all by the sum
  For i = 1 To n_pop
```

```
cum_fitness(i) = cum_fitness(i) / cum_fitness(n_pop)
Next i

select_pairs pairs, cum_fitness
'cross them over
cross_over_genes pairs
'and mutate
mutate_genes
frame_shift_genes
'and if we're saving the best
If save_best = True Then
 For i = 1 To n_genes
  genome(i, n_pop) = saved_genome(i)
 Next i
End If
End Sub
Private Sub frame_shift_genes()
'select which genes to frame shift
'for these randomly select two points,
'in the first place
 Dim this_gene As Integer, this_ind As Integer
 Dim start As Integer, last As Integer
 Dim rand As Single

  For this_ind = 1 To UBound(genome, 2)
   rand = Rnd()
   If rand < frame_shift_prob Then
    start = Rnd() * UBound(genome, 1) + 1
    last = Rnd() * UBound(genome, 1) + 1
    If last > start Then
    If last >= UBound(genome, 1) Then last = UBound(genome, 1) - 1
    If start <= 1 Then start = 1
       For this_gene = start To last - 1
       genome(this_gene + 1, this_ind) = genome(this_gene, this_ind)
     Next this_gene
     genome(last, this_ind) = genome(start, this_ind)
    End If
    If start > last Then
    'last can't be 1 or you try to write to position 0
    If last <= 1 Then last = 2
    If start >= UBound(genome, 1) Then start = UBound(genome, 1)
     For this_gene = start To last Step -1
      genome(this_gene - 1, this_ind) = genome(this_gene, this_ind)
     Next this_gene
     genome(last, this_ind) = genome(start, this_ind)
    End If
   End If 'rand < frame_shift if
  Next this_ind
End Sub

Private Sub cross_over_genes(pairs() As Integer)
'create new genome
'there are n_pop/2 pairs. each pair results in 2 individuals in the new_genome
Dim new_genome() As Boolean
Dim this_pair As Integer, n_pairs As Integer, length As Integer
Dim this_gene As Integer '
```

```
' genome is (gene,sub)
' pairs is (1 to 2,nsub/2)
n_pairs = UBound(pairs, 2)
length = UBound(genome, 1)
ReDim new_genome(length, n_pairs * 2)
Dim where As Integer, rand As Single
For this_pair = 1 To n_pairs
' new individuals are (this_pair-1)*2 +1 and this_pair *2 in new_genome
 If Rnd() < cross_over_freq Then
  where = Rnd() * length
  For this_gene = 1 To where
   ' write the left half of the gene, up to where
   new_genome(this_gene, (this_pair - 1) * 2 + 1) = genome(this_gene, pairs(1, this_pair))
   new_genome(this_gene, this_pair * 2) = genome(this_gene, pairs(2, this_pair))
  Next this_gene
  For this_gene = where + 1 To length
   new_genome(this_gene, (this_pair - 1) * 2 + 1) = genome(this_gene, pairs(2, this_pair))
   new_genome(this_gene, this_pair * 2) = genome(this_gene, pairs(1, this_pair))
  Next this_gene
 Else
  ' no cross over
  For this_gene = 1 To length
   new_genome(this_gene, (this_pair - 1) * 2 + 1) = genome(this_gene, pairs(1, this_pair))
   new_genome(this_gene, this_pair * 2) = genome(this_gene, pairs(2, this_pair))
  Next this_gene
 End If
Next this_pair
' then copy new_genome to genome
For this_gene = 1 To length
 For this_pair = 1 To n_pairs * 2
 genome(this_gene, this_pair) = new_genome(this_gene, this_pair)
 Next this_pair
Next this_gene
End Sub

Private Sub mutate_genes()
Dim n_genes As Integer, this_gene As Integer
Dim n_pop As Integer, this_ind As Integer
Dim rand As Single
n_genes = UBound(genome, 1): n_pop = UBound(genome, 2)
For this_gene = 1 To n_genes
 For this_ind = 1 To n_pop
 rand = Rnd()
 If rand < mutation_rate Then genome(this_gene, this_ind) = Not (genome(this_gene, this_ind))
 Next this_ind
Next this_gene
End Sub
Function get_nbits(number As Integer) As Integer
If number = 0 Then
get_nbits = 0
Else
get_nbits = Log(number) / Log(2) + 0.4999
End If
End Function
Sub run_population(scaled_fitness() As Single, values() As Integer, check_out As Boolean)
Dim limit_str As String  ' is theta at the upper or lower limit
```

```
Dim genome_integer As Double, old_fitness As Single
Dim old_gen As Integer, old_ind As Integer, old_dir As String
Dim already_run As Boolean
Dim fitness() As Single
Dim n_ind As Integer, control_code As String, i As Integer, ok As String
Dim n_genes As Integer, one_run_values() As Integer
Dim obj As Single, success As Integer, covar As Integer
n_ind = UBound(values, 2)
ReDim fitness(n_ind)
ReDim new_fitness(n_ind)
n_genes = UBound(values, 1)
ReDim one_run_values(1 To n_genes)
Dim this_runl As Integer
this_run = 0
frm_main.pgb_ind.max = n_ind
'*****************************************************************
'*************** TOP OF POPULATION LOOP ************************
'*****************************************************************
For this_runl = 1 To n_ind ' this_runl = local this_run
frm_main.pgb_ind.value = this_runl
frm_main.Refresh
DoEvents
Do While paused = True
 DoEvents
 Sleep 500
Loop
this_run = this_run + 1
frm_main.spr_result.col = 9
frm_main.spr_result.row = (this_gen - 1) * n_ind + this_runl
frm_main.spr_result.value = this_runl
frm_main.spr_result.col = 8
frm_main.spr_result.value = this_gen
DoEvents
If stop_run = True Then Exit Sub
 For i = 1 To n_genes
  one_run_values(i) = values(i, this_runl)
 Next i
 control_code = make_control(frm_main.txt_code, one_run_values(), token_collection())
 ok = Dir(gen_directory & "\" & this_runl, vbDirectory)
 If Trim(ok) = Trim(str(this_runl)) Then
  ok = Dir(gen_directory & "\" & this_runl & "\control", vbDirectory)
  If ok <> "" Then Kill gen_directory & "\" & this_runl & "\*.*"
 Else
  MkDir gen_directory & "\" & this_runl
  ChDir gen_directory & "\" & this_runl
 End If
 Open gen_directory & "\" & this_runl & "\control" For Output As #1
 Print #1, control_code
 Close #1
 ' first check to see if this genome has been run
 already_run = False
 limit_str = "" ' reset the string that describes whether theta is at the boundry to null
 genome_integer = make_int(this_runl)
 For i = 1 To n_unique
  If genome_integer = unique_fit(i, 1) Then
   fitness(this_runl) = unique_fit(i, 2)
```

```
  old_gen = unique_fit(i, 3)
  old_ind = unique_fit(i, 4)
  obj = unique_fit(i, 5)
  success = unique_fit(i, 6)
  covar = unique_fit(i, 7)
 run_number = run_number + 1
  already_run = True
  ' need to recalcuate limit_str for already run model NOT DONE YET
  Exit For
 End If

Next i
Sleep 100 ' to clear file buffer before deleteing files.
If already_run = False Then
 fitness(this_runl) = call_nm("c:", gen_directory & "\" & this_runl, "control", obj, success, covar, limit_str,
check_out)
On Error Resume Next ' don't ned to worry if you can't delete file
If Dir("fdata", vbNormal) <> "" Then Kill ("fdata")
If Dir("link.lnk", vbNormal) <> "" Then Kill ("link.lnk")
If Dir("prderr", vbNormal) <> "" Then Kill ("prderr")
If Dir("freport", vbNormal) <> "" Then Kill ("freport")
If Dir("fsubs", vbNormal) <> "" Then Kill ("fsubs")
If Dir("fsubs.for", vbNormal) <> "" Then Kill ("fsubs.for")
If Dir("nonmem.exe", vbNormal) <> "" Then Kill ("nonmem.exe")
If Dir("df.txt", vbNormal) <> "" Then Kill ("df.txt")
On Error GoTo 0
 n_unique = n_unique + 1
 frm_main.lbl_count = n_unique
 unique_fit(n_unique, 1) = genome_integer: unique_fit(n_unique, 2) = fitness(this_runl)
 unique_fit(n_unique, 3) = this_gen: unique_fit(n_unique, 4) = this_runl
 unique_fit(n_unique, 5) = obj: unique_fit(n_unique, 6) = success
 unique_fit(n_unique, 7) = covar
 ' need to write the results to the spreadsheet, usually done by call_nm
 Else
 old_dir = home_directory & "\" & Trim(str(old_gen)) & "\" & Trim(str(old_ind)) & "\"
 ChDir (gen_directory & "\" & this_runl)
 If LCase(Dir(old_dir & "control")) = "control" Then
  FileCopy old_dir & "control", CurDir & "\control"
 End If
 If LCase(Dir(old_dir & "result")) = "result" Then
  FileCopy old_dir & "result", CurDir & "\result"
 End If
 If LCase(Dir(old_dir & "output")) = "output" Then
  FileCopy old_dir & "result", CurDir & "\output"
 End If
 If LCase(Dir(old_dir & "inputs")) = "inputs" Then
  FileCopy old_dir & "inputs", CurDir & "\inputs"
 End If
 If LCase(Dir(old_dir & "parms")) = "parms" Then
  FileCopy old_dir & "parms", CurDir & "\parms"
 End If
 End If
 ' and write the results
 With frm_main.spr_result
   .col = 2
   If success = 0 Then
```

```
    .text = "Yes"
    Else
    .text = "No"
    End If
    .row = run_number
    .col = 3
    If success = 0 And covar = 0 Then
    .text = "Yes"
    Else
    .text = "No"
    End If
    .col = 4
    .text = fitness(this_runl)
    .col = 1
    If obj < 999999999.9 Then
    .text = obj
    Else
    .text = "Crash"
    .col = 2
    .text = "No"
    .col = 3
    .text = "No"
    .col = 4
    .text = "Crash"
    End If
    .col = 11
    .text = limit_str
    .col = 1
    .Action = 0
End With
DoEvents: frm_main.Refresh
If save_control = False Then Kill "control"
If save_output = False Then
  If Dir(".\output", vbNormal) = "result" Then Kill "result"
End If
If stop_run = True Then Exit Sub
Next this_runl
scale_fitness scaled_fitness(), fitness()
' update plot
update_plot fitness(), scaled_fitness()
End Sub
Sub update_plot(fitness() As Single, scaled_fitness() As Single)
' first append new fitness values onto all_fitness
' need to check to see if n generations is exceeded for time limited
Dim i As Integer, n As Integer
Dim this_min As Single, this_max As Single, this_mean As Single
Dim sum As Single, count As Integer
this_min = 999999999
this_max = -99999999
sum = 0
count = 0
For i = 1 To UBound(fitness)
 If fitness(i) < 9999999 Then
    If fitness(i) < this_min Then this_min = fitness(i)
    If fitness(i) > this_max Then this_max = fitness(i)
    sum = sum + fitness(i)
```

```
  count = count + 1
 End If
Next i
If count <> 0 Then
this_mean = sum / count
Else
this_mean = 1000000
End If
With frm_main.MSChart1
 .row = this_gen
.Column = 1
.Data = this_min
.Column = 2
.Data = this_mean
.Column = 3
.Data = this_max
.DrawMode = VtChDrawModeDraw
End With

With frm_main.spr_result
.col = 10
 For i = 1 To pop_size
 .row = (this_gen - 1) * pop_size + i
 .text = Format(scaled_fitness(i), "0.000")
 Next i


End With
End Sub
Sub initialize_plot(n_rows As Integer)
Dim i As Integer, n As Integer
'final 2 columns define upper limit of axis
With frm_main.MSChart1
.RowCount = 0
.ColumnCount = 0
.RowCount = n_rows
.ColumnCount = 3
For i = 1 To n_rows
  .row = i
  .RowLabel = i
Next i
   .DrawMode = VtChDrawModeDraw
   End With
End Sub
Function get_n_pop() As Double
Dim this_group As Integer, n_sets As Double
n_sets = 1
For this_group = 1 To n_token_groups
n_sets = n_sets * token_collection(this_group).n_token_sets
Next this_group
get_n_pop = n_sets
End Function
Function make_control(ga_code As String, values() As Integer, _
          token_collection() As token_group) As String
'so, search ga_code for each instance of stem(1) to stem(n_token_groups)
'thetas will be "{THETA(a)}" in the $PK, ERROR or pred and "{$THETA(A) =} (0,1,2) for the theta
```

```
'part.  similarly for omega and sigma
'afterward, well need to sort out A, B etc and put the $THETA, $OMEGA and $SIMGA
'element in proper order
'and mover the {$THETA(A)= } TO AFTER THE VALUES
Dim done As Boolean, this_int As Integer
'If this_run > 15 Then
'MsgBox "Pause"
'End If
Dim this_token_set As Integer
Dim this_token As Integer
Dim new_code As String, new_string As String, old_string As String
new_code = ga_code
Dim test_string As String
done = False
While Not (done) 'loop here until no more token_stem(*) is found
'this will loop trough the non-omega genes
 For this_token_set = 1 To n_token_groups
  For this_token = 1 To token_collection(this_token_set).n_tokens
    old_string = token_collection(this_token_set).stem & "(" & this_token & ")"
    new_string = token_collection(this_token_set).get_token(values(this_token_set), this_token)
    new_code = sub_string(new_code, old_string, new_string)
'    frm_text.txt_text = new_code
'    frm_text.Show 1, frm_main
  Next this_token
 Next this_token_set
 'check to see if we are done
 'loop over token_set stem to look for more tokens
 'look for token_set.stem & (1-9)
 done = True
 For this_token_set = 1 To n_token_groups
  For this_int = 1 To max_theta
  test_string = token_collection(this_token_set).stem & "(" & Trim(str(this_int)) & ")"
   ' if test_string is in code, then not done
   frm_text.txt_text = new_code
   'frm_text.Show 1, frm_main
   If InStr(1, new_code, test_string, vbTextCompare) <> 0 Then
   done = False
    Exit For
    End If
  Next this_int
  If done = False Then Exit For
 Next this_token_set
Wend 'end of loop over
'now the final editting, replace the {crlf} with real crlf
   new_code = add_crlf(new_code)
'If MsgBox(new_code, vbOKCancel, "before match_reference") = vbCancel Then End
   'match up the THETA(A) with correct THETA(1)'s
   new_code = match_references(new_code)
   'sort the resulting theta, etas, sigmas
   new_code = sorter(new_code)
'finally, swap $THETA= to end of line
   new_code = swapper(new_code)
'If MsgBox(new_code, vbOKCancel, "final") = vbCancel Then End
'now, if omega_block is true, first remove the all omega parts and substitute the omega BLOCK(n) parts
If omega_block = True Then
'sustitute the BLOCK syngtax
```

```
'frm_text.txt_text = new_code
'frm_text.Show 1, frm_main
 new_code = sub_omega_block(new_code, values)
End If
make_control = new_code
'MsgBox new_code
End Function
Function sub_omega_block(code As String, values() As Integer) As String
Dim block_part As String, n_etas As Integer
Dim omega_start As Integer, n As Integer
Dim omega_end As Integer, this_gene As Integer
Dim start_pos As Integer, end_pos As Integer
Dim init_omega() As Single: ReDim init_omega(1 To n_omega)
Dim sequences() As Integer: ReDim sequences(1 To n_omega - 1) ' sequence of omegas, all possible etas
(ie max omega, n_omega)
Dim new_sequences() As Integer ' the sequences specefic for this control
                    ' sequence will have max_omega values
Dim left_part_code As String, right_part_code As String  ' left and right parts of code, without the omega
block
Dim covar_values() As Boolean: ReDim covar_values(1 To n_omega - 1) ' is this row in a block with the
row above? does not include first row
 ' read in covar_values() go ahead an read in all , even though we'll only use some
For this_gene = 1 To n_omega - 1
 If values(this_gene + n_non_omega_genes) = 2 Then
  covar_values(this_gene) = True
 End If
Next this_gene
 ' while we're here, read in sequences
For this_gene = 1 To n_omega - 1
 sequences(this_gene) = values(this_gene + n_non_omega_genes + n_omega - 1)

 ' while we're here, read in sequences
Next this_gene
 'we'll need to compress the sequence of the first n sequence values into 1 to n
 ' e.g. if we have 4 etas in this control, but max_omega = 7, and sequence is 5,6,3,1,24
 ' we compress the first 4 into 3,4,2,1
 ' so we need to figure out how many etas in this control file.
n_etas = count_etan(code)
 'frm_text.txt_text = code
 'frm_text.Show 1, frm_main
 ' first cut out the omega block
omega_start = InStr(1, code, "$OMEGA")
omega_end = InStr(1, code, "$SIGMA") - 1
left_part_code = Left(code, omega_start - 1)
'frm_text.txt_text = left_part_code
'frm_text.Show 1, frm_main
right_part_code = Right(code, Len(code) - omega_end)
'frm_text.txt_text = right_part_code
'frm_text.Show 1, frm_main
block_part = Mid(code, omega_start, omega_end - omega_start)
'frm_text.txt_text = block_part
'frm_text.Show 1, frm_main
 ' remove $OMEGA
 block_part = Right(block_part, Len(block_part) - 7) ' 7 for the $OMEGA
'frm_text.txt_text = block_part
'frm_text.Show 1, frm_main
```

```
'remove all parts between ; and vbcrlf
While InStr(1, block_part, ";") <> 0
' If this_run = 4 Then
'  frm_text.txt_text = block_part
'  frm_text.Show 1, frm_main
' End If
 start_pos = InStr(1, block_part, ";") - 1
 'find end of line
 end_pos = InStr(start_pos, block_part, vbCrLf)
 If end_pos = 0 Then end_pos = Len(block_part)
'MsgBox Left(block_part, start_pos)
 block_part = Left(block_part, start_pos) & Right(block_part, Len(block_part) - end_pos)
 'MsgBox block_part
Wend
block_part = Trim(block_part)
'now get the values for omega' just read the (??) values
'find pairs between ( and ) and read into init_omega
start_pos = 1: n = 1
While InStr(start_pos, block_part, "(") <> 0
 start_pos = InStr(start_pos, block_part, "(") + 1
 end_pos = InStr(start_pos, block_part, ")")
 init_omega(n) = Val(Mid(block_part, start_pos, end_pos - start_pos))
  n = n + 1
  Wend
'and resequence them
 'if there are no etas exit, will cause a crash in nonmem, but that's ok
 If n_etas > 0 Then
 ReDim new_sequences(1 To n_etas)
 Else
 sub_omega_block = code
 Exit Function
 End If
 'read in the new sequences from the sequenes
 'loop over sequences, looking for values 1 to n_etas.
 'note that sequences has n_omega -1 elements, the final
 'posiiton is determined by the others
 Dim cur_eta_count As Integer ' count of non-zero elements
 Dim n_etas_left As Integer ' how many etas are left to fill
 Dim i As Integer, cur_eta_position 'current position in new_sequences being examined (to see if = 0)
 n_etas_left = n_etas
 For i = 1 To n_etas 'looking for ETA(i) in main code
 'get the values from sequences
  cur_eta_position = sequences(i)
  If cur_eta_position > n_etas_left Then cur_eta_position = n_etas_left
  cur_eta_count = 0
  For n = 1 To n_etas
   If new_sequences(n) = 0 Then cur_eta_count = cur_eta_count + 1
   If cur_eta_count = cur_eta_position Then
     new_sequences(n) = i
     n_etas_left = n_etas_left - 1
     Exit For
   End If
 Next n
Next i
' substitute the etas
' first change THETA to XXXXX
```

```
'frm_text.txt_text = code
'frm_text.Show 1, frm_main
left_part_code = sub_string(left_part_code, "THETA", "XXXXX")
Dim new_string As String, old_string As String 'new string will be "YYY(" to prevent re- replacement
For i = 1 To n_etas
 old_string = "ETA(" & Trim(str(i)) & ")"
 new_string = "YYY(" & Trim(new_sequences(i)) & ")"
  left_part_code = sub_string(left_part_code, old_string, new_string)
Next i
 'and replace the "YYY(" with "ETA("
left_part_code = sub_string(left_part_code, "YYY(", "ETA(")
left_part_code = sub_string(left_part_code, "XXXXX", "THETA")
'frm_text.txt_text = left_part_code
'frm_text.Show 1, frm_main
'and write the new block
'build new omega block
Dim new_omega_block As String, this_column As Integer
Dim this_row As Integer, cur_end_row As Integer
Dim cur_row_count 'how many rows in this block
Dim off_diag As String
  this_column = 1: this_row = 1
cur_end_row = 1
While cur_end_row <= n_etas
 'is this a new block, if so how big?
 'loop through covar_values until you find a false
 cur_row_count = 1
 '**************************************
 '***********Part to put into old project
 '**************************************
 Do While covar_values(cur_end_row) And cur_end_row < n_etas
  cur_row_count = cur_row_count + 1
  cur_end_row = cur_end_row + 1
  If cur_end_row > n_etas Then Exit Do
 Loop
 'need to see if we have exceeded the number of etas
 If cur_end_row > n_etas Then cur_end_row = n_etas
 If cur_row_count = 1 Then
  new_omega_block = new_omega_block & "$OMEGA " & vbCrLf
  Else
  new_omega_block = new_omega_block & "$OMEGA BLOCK(" & Trim(str(cur_row_count)) & ")" &
vbCrLf
 End If
 For i = 1 To cur_row_count
 'construct off diagonal elements
  off_diag = " "
  For n = 1 To i - 1
   off_diag = off_diag & " (0.00001) "
  Next n
  new_omega_block = new_omega_block & off_diag & "(" & init_omega(1) & ")" & vbCrLf
 Next i
'MsgBox new_omega_block
 cur_end_row = cur_end_row + 1
Wend 'this_row
sub_omega_block = left_part_code & new_omega_block & right_part_code

'frm_text.txt_text = left_part_code & vbCrLf & new_omega_block & vbCrLf & right_part_code
```

```
'frm_text.Show 1, frm_main
 End Function

' and sequence the etas
 Function sequence_omegas(code As String)
 sequence_omegas = code
 End Function
Function sorter(ByVal code As String) As String
' this function sorts the theta,omegas and sigma initial estimates
Dim new_code As String, cut_out As String, this_prefix As Integer
Dim stack(1 To max_theta) As String, stack_order(1 To max_theta) As Integer
Dim new_cut_out As String, token As String
Dim first_position As Long, last_position As Long, next_position As Long
Dim cur_prefix As String, position As Long
Dim prefixes(1 To 3) As String, stack_position As Integer
prefixes(1) = "{$THETA(": prefixes(2) = "{$ETA(": prefixes(3) = "{$EPS("
'If MsgBox(code, vbOKCancel, "in sorter") = vbCancel Then End
Dim old_cut_out As String ' need to preservie original cut out to use in sub_string
new_code = code
For this_prefix = 1 To 3
'If MsgBox(code, vbOKCancel) = vbCancel Then End
 If InStr(new_code, prefixes(this_prefix)) = 0 Then Exit For
 'MsgBox new_code
 'collect all {THETA(?)=} (XXX)}
 ' find the first {THETA
 first_position = InStr(1, new_code, prefixes(this_prefix))
 last_position = first_position
 ' then find end of theta section
 next_position = InStr(last_position + 1, new_code, prefixes(this_prefix))
 While next_position <> 0
  next_position = InStr(last_position + 1, new_code, prefixes(this_prefix))
  If next_position <> 0 Then last_position = next_position
 Wend
 ' then find the end of the last theta string
 ' note that you must end with a ")"
 ' find the first "}" at the end of the string
 last_position = InStr(last_position, new_code, "}")
 ' then the final ")"
 last_position = InStr(last_position - 1, new_code, ")") + 1
 ' cut out that section and sort it
 'MsgBox new_code
 cut_out = Mid(new_code, first_position, last_position - first_position)
 old_cut_out = cut_out 'need to save old cut_out for substring, since we are about
 ' to  change cut_out.
 ' remove all vbcrlf from cut_out
 cut_out = sub_string(cut_out, vbCrLf, "")
 'MsgBox cut_out
 'assemble the stack of values
 ' find the lowest value, put it in stack(1) etc,
 Dim i As Integer, cur_start As String, n As Integer, token1 As String, token2 As String, token3 As String
  For i = 1 To max_theta
   cur_start = prefixes(this_prefix) & i & ")=}"
   first_position = InStr(1, cut_out, cur_start)
   ' seperate each token (from $theta to before next $theta) and put then on a stack to be sorted.
  If first_position <> 0 Then
   stack_position = stack_position + 1 ' next stack position
```

```
    'find the start of the next token, or the end of cut_out string
    last_position = InStr(first_position + 5, cut_out, "{$") - 1 '
    If last_position < 0 Then last_position = Len(cut_out)
    token = Mid(cut_out, first_position, last_position - first_position + 1)
'   token1 = Mid(cut_out, first_position + Len(cur_start), _
'   last_position - first_position - Len(cur_start))
'   token2 = "  ;" & Mid(cut_out, first_position, Len(cur_start))
    stack(stack_position) = token
    stack_order(stack_position) = i
   End If
  Next i
  'put cut out back together
  new_cut_out = ""
   For i = 1 To stack_position
    new_cut_out = new_cut_out & stack(i) & vbCrLf & " "
   Next i
    new_code = sub_string(new_code, old_cut_out, new_cut_out)
   new_cut_out = ""
  'MsgBox Mid(new_code, 500, Len(new_code) - 500)
   stack_position = 0
  Next this_prefix
sorter = new_code
End Function

Private Function swapper(code As String) As String
'this function puts the {$THETA(?)} after the value
Dim this_prefix As Integer, position As Integer, eol As Integer, stop_pos As Integer
Dim cut_out As String, rest_str As String, new_str As String, first_part As String
Dim prefixes(1 To 3) As String: prefixes(1) = "{$THETA(": prefixes(2) = "{$ETA(": prefixes(3) =
"{$EPS(":
For this_prefix = 1 To 3
'loop over the text looking for prefix
position = InStr(1, code, prefixes(this_prefix))
While position <> 0
'now find end of line
 stop_pos = InStr(position, code, "}")
 eol = InStr(stop_pos, code, vbCr) - 1
 cut_out = Mid(code, position, eol - position + 1) '+ 1)
 stop_pos = InStr(position, code, "}")
 rest_str = Trim(Mid(code, stop_pos + 1, eol - stop_pos))
 first_part = Trim(Mid(code, position, stop_pos - position))
 Mid(first_part, 1, 2) = ";;"
 new_str = rest_str & first_part
 code = sub_string(code, cut_out, new_str)
 position = eol - 3
 position = InStr(1, code, prefixes(this_prefix))
Wend
Next this_prefix
swapper = code
End Function
Function match_references(ByVal code As String)
'match up {THETA(A) with {THETA(A) =}(xxxx) to figure out which theta (eta) this is
'and ETA(A) with {ETA(A) =} XX
'and also sigma
Dim this_prefix As Integer, integer_used As Boolean
Dim position As Long, cur_prefix As String, next_value As Integer
```

```
Dim cur_letter As Integer, cur_integer As Integer, cur_string As String
Dim cur_new_string As String, cur_old_string As String
Dim vtheta_used As Boolean
' first pass through the data to find out which theta, eta and eps is available
' Do theta first note that ETA is substring of THETA
' ETA is in THETA, so we'll first change "THETA" to "XXXXX", do eta then change back
code = sub_string(code, "THETA", "XXXXX")
'   MsgBox (code)
Dim prefixes(1 To 3) As String: prefixes(1) = "ETA": prefixes(2) = "THETA": prefixes(3) = "EPS"
For this_prefix = 1 To 3
 cur_prefix = prefixes(this_prefix)
 ' find out if there are any fixed thetas (i.e., theta(1))
'frm_text.txt_text = code
'frm_text.Show 1, frm_main

' MsgBox (code)
 cur_string = cur_prefix & "(1)"
 position = InStr(code, cur_string)
 cur_integer = 1
 ' find first available theta value
 While position > 0
   cur_integer = cur_integer + 1
   cur_string = cur_prefix & "(" & cur_integer & ")"
   position = InStr(code, cur_string)
 Wend
 ' assign lowest available number
 next_value = cur_integer
 ' MAIN LOOP THROUGH CODE TO SUBSTITUTE THETA(3) FOR THETA(A)
 ' now loop through each posible value to variable theta (theta(a) to theta(z)
 ' NEED MORE LETTER THAN 26
 For cur_letter = Asc("A") To Asc("Z")
 vtheta_used = False
  ' get new theta string
   cur_new_string = cur_prefix & "(" & cur_integer & ")"
  ' get old variable theta string
   cur_old_string = "{" & cur_prefix & "(" & Chr(cur_letter) & ")}"
   position = InStr(1, code, cur_old_string)
   While position > 0
    vtheta_used = True
    code = sub_string(code, cur_old_string, cur_new_string)
   '  MsgBox code
    position = InStr(1, code, cur_old_string)
   Wend
    ' now do {theta(A)=} part
    ' right now we'll just replace, will need to sort and put theta(1) part
    ' at the end later.
   cur_new_string = "{$" & cur_prefix & "(" & cur_integer & ")=}"
   cur_old_string = "{$" & cur_prefix & "(" & Chr(cur_letter) & ")=}"
   position = InStr(1, code, cur_old_string)
   While position > 0
    code = sub_string(code, cur_old_string, cur_new_string)
    position = InStr(1, code, cur_old_string)
   Wend
   ' we want to increment cur_integer only if variable theta is used
   If vtheta_used = True Then cur_integer = cur_integer + 1
 Next cur_letter
```

```
'NOW THETA(AA) TO THETA(AZ)

For cur_letter = Asc("A") To Asc("Z")
vtheta_used = False
 'get new theta string
  cur_new_string = cur_prefix & "(" & cur_integer & ")"
 'get old variable theta string, WITH THE "A"
  cur_old_string = "{" & cur_prefix & "(A" & Chr(cur_letter) & ")}"
  position = InStr(1, code, cur_old_string)
  While position > 0
    vtheta_used = True
    code = sub_string(code, cur_old_string, cur_new_string)
   ' MsgBox code
    position = InStr(1, code, cur_old_string)
  Wend
   'now do {theta(AA)=} part
   'right now we'll just replace, will need to sort and put theta(1) part
   'at the end later.
  cur_new_string = "{$" & cur_prefix & "(" & cur_integer & ")=}"
  cur_old_string = "{$" & cur_prefix & "(A" & Chr(cur_letter) & ")=}"
  position = InStr(1, code, cur_old_string)
  While position > 0
    code = sub_string(code, cur_old_string, cur_new_string)
    position = InStr(1, code, cur_old_string)
  Wend
  'we want to increment cur_integer only if variable theta is used
  If vtheta_used = True Then cur_integer = cur_integer + 1
Next cur_letter


'NEXT PREFIX
'change XXXX back to THETA if we just did theta
If this_prefix = 1 Then code = sub_string(code, "XXXXX", "THETA")
Next this_prefix
'If MsgBox(code, vbOKCancel, "end of match_ref") = vbCancel Then End
match_references = code
End Function

Function add_crlf(ByVal code As String) As String
'replace the {crlf} with vbcrlf
Dim where As Long
where = InStr(1, code, "{crlf}")
While where > 0
code = sub_string(code, "{crlf}", vbCrLf)
where = InStr(1, code, "{crlf}")
Wend
add_crlf = code
End Function

Function sub_string(code As String, old_str As String, new_str As String) As String
'first check to see if code has changed
If InStr(new_str, old_str) > 0 Then
  sub_string = code
Else
 'this function replaces all instances of old_str with new_str
  Dim position As Long, new_code As String, left_part As String, right_part As String
```

```
 position = InStr(1, UCase(code), UCase(old_str)) - 1
 new_code = code
 While position > 0
  left_part = Left(new_code, position)
  right_part = Right(new_code, Len(new_code) - position - Len(old_str))

  new_code = left_part & new_str & right_part
  position = InStr(position, new_code, old_str) - 1
 Wend
'MsgBox (sub_string)
'MsgBox (new_str)
'MsgBox (old_str)
 sub_string = new_code
End If
'MsgBox (sub_string)
End Function


Function count_non_omega_genes()
count_non_omega_genes = n_token_groups
End Function
Function count_omega_genes()
If omega_block = True Then
n_omega = count_max_omega
count_omega_genes = 2 * (n_omega - 1)
' n_omega genes for the sequence (n_omega-1)! and n_omega -1 for the block definition
Else
 count_omega_genes = 0
End If
End Function
Function count_non_omega_bits()
Dim i As Integer
Dim n As Integer
For i = 1 To n_token_groups
' one token set requires no genes, two requres 1, 3 or 4 requires 2, 5 to 8 requires 3
' basically, ceiling(log base 2(n_token_sets)
n = n + CInt(Log(token_collection(i).n_token_sets) / Log(2) + 0.499999)
Next i
count_non_omega_bits = n
End Function
Function count_omega_bits()
Dim i As Integer
Dim n As Integer
If omega_block = True Then
 n = n_omega - 1  ' for the block part, one bit per gene (is this in a block with the row above?)
  ' sequence part (n-1)!
  For i = n_omega To 2 Step -1
   ' eta(1) can be in n_omega possible values, eta(2) can be in n_omega -1 etc,
   ' to eta(n_omega) which is fixed
   n = n + CInt(Log(i) / Log(2) + 0.499999)
  Next i
  count_omega_bits = n
 Else
 count_omega_bits = 0
 End If
 End Function
```

```
Sub randomizer(ByRef genome() As Boolean)
Dim i As Integer, n As Integer
Randomize
For n = 1 To pop_size
  For i = 1 To UBound(genome, 2)
    genome(n, i) = CInt(Rnd())
  Next i
Next n
End Sub


Public Sub save_model(file_name As String)
' get file name
' need to save:
' control file
' genome
' token set
' options
'first, control file
n_models = 1
Dim i As Integer, n As Integer, p As Integer
file_name = home_directory & "\" & file_name
Open file_name For Output As #1
Print #1, "Number of models = " & vbCrLf & n_models
For i = 1 To n_models
Print #1, "########## Begining of model # " & i & " ############"
Print #1, frm_main.txt_code
Print #1, "########## End of model # " & i & " ################"
Next i
Print #1, "### End of GA code ###"
Print #1, " Last gen", vbCrLf, last_gen
Print #1, "### Start of genome ###"
Dim gen_str As String
If run_number = 0 Then
Print #1, "Genome not defined"
Else
Dim n_bits As Integer
n_bits = count_omega_bits() + count_non_omega_bits()
Print #1, "N bits = ", vbCrLf, UBound(genome, 1)
Print #1, "Pop size =  ", vbCrLf, UBound(genome, 2)
' genome is n_bits by pop size
 For i = 1 To UBound(genome, 1)
   For n = 1 To UBound(genome, 2)
     gen_str = gen_str & " " & -Int(genome(i, n)) ' - because fals = 0, true = -1
   Next n
     Print #1, gen_str
     gen_str = ""
 Next i
 End If
 Print #1, "### End of genome ###"
 Print #1, "### Begining of tokens ###"
 For i = 1 To n_token_groups
 Print #1, "Group stem =    ", vbCrLf, token_collection(i).stem
 Print #1, "N token sets =  ", vbCrLf, token_collection(i).n_token_sets
 Print #1, "N tokens =      ", vbCrLf, token_collection(i).n_tokens
```

```
 For n = 1 To token_collection(i).n_token_sets
Print #1, "Token set # ", vbCrLf, n
  For p = 1 To token_collection(i).n_tokens
  Print #1, token_collection(i).get_token(n, p)
  Next p
 Next n
Next i

Print #1, "### End of tokens ###"
Print #1, "### Begining of options ###"
Print #1, "cross_over_freq", vbCrLf, cross_over_freq
Print #1, "mutation_rate", vbCrLf, mutation_rate
Print #1, "frame shift prob", vbCrLf, frame_shift_prob
Print #1, "n_runs", vbCrLf, n_runs
Print #1, "theta_crit", vbCrLf, theta_crit
Print #1, "omega_crit", vbCrLf, omega_crit
Print #1, "sigma_crit", vbCrLf, sigma_crit
Print #1, "cov_crit", vbCrLf, cov_crit
Print #1, "pop_size", vbCrLf, pop_size
Print #1, "generation_limit", vbCrLf, generation_limit
Print #1, "call_method", vbCrLf, call_method
Print #1, "upper fitness limit", vbCrLf, upper_fitness_limit
Print #1, "lower_fitness_limit", vbCrLf, lower_fitness_limit
Print #1, "correlation criteria", vbCrLf, corr_crit
Print #1, "success_criteria", vbCrLf, success_crit
Print #1, "save control", vbCrLf, save_control
Print #1, "save best", vbCrLf, save_best
Print #1, "save output", vbCrLf, save_output
Print #1, "omega block", vbCrLf, omega_block
Print #1, "seed type", vbCrLf, seed_type
Print #1, "seed value", vbCrLf, seed_value
Print #1, "### End of options ###"
Print #1, "######################"
Close #1
End Sub
Sub get_model(file As String)
Dim textline As String, code As String, n As Integer, n_bits As Integer
Dim n_token_sets As Integer
Dim i As Integer, token_set_num As Integer, token_num As Integer
For i = 1 To n_token_groups
token_collection(i).clear
Next i
n_token_groups = 0
If Dir(file) = "" Then
  MsgBox ("File not found")
  Exit Sub
End If
' 'see if it is on the start_files
' For i = 1 To n_files
'  If start_files(i) = file Then
'    For n = i To n_files - 1
'      start_files(i) = start_files(i + 1)
'      frm_main.files(i).Caption = start_files(i)
'    Next n
'    start_files(n_files) = ""
'    frm_main.files(n_files).Visible = False
```

```
'    n_files = n_files - 1
'  End If
'
' Next i
'************************


For i = 1 To n_files
  If start_files(i) = file Then
'remove it
    For n = i To n_files - 1 Step 1
      start_files(n) = start_files(n + 1)
    Next n
  n_files = n_files - 1
    start_files(n_files + 1) = ""
    Exit For
  End If
Next i
If n_files < 4 Then n_files = n_files + 1
For i = n_files To 2 Step -1
start_files(i) = start_files(i - 1)
frm_main.files(i).Caption = start_files(i)
Next i
start_files(1) = file
frm_main.files(1).Caption = start_files(1)
 '***************




Open file For Input As #1

Line Input #1, textline 'Number of models =
Line Input #1, textline ' number of models
n_models = Val(textline)
Line Input #1, textline ' number of models

code = ""
For i = 1 To n_models
Line Input #1, textline ' number of models
While Left(textline, 24) <> "############ End of model"
 code = code & textline & vbCrLf
 Line Input #1, textline
Wend
frm_main.txt_code = code
Next i
Line Input #1, textline '### end of ga code ###
Line Input #1, textline 'last_gen
Line Input #1, textline
last_gen = Val(textline)
Line Input #1, textline '### Start of genome ###
Line Input #1, textline
If Trim(textline) <> "Genome not defined" Then
Line Input #1, textline
```

```
n_bits = Val(textline)
Line Input #1, textline
Line Input #1, textline
pop_size = Val(textline)
Dim value As Integer
ReDim genome(1 To n_bits, 1 To pop_size)
 For i = 1 To n_bits
  For n = 1 To pop_size
   Input #1, value
   genome(i, n) = Val(value)
  Next n
Next i
End If

Line Input #1, textline '### Begining of tokens ###
While Trim(textline) <> "### Begining of tokens ###"
  Line Input #1, textline
Wend
While Trim(textline) <> "### End of tokens ###"

 While Trim(textline) <> "Group stem ="
  Line Input #1, textline
 Wend
 Line Input #1, textline
  n_token_groups = n_token_groups + 1
  token_collection(n_token_groups).stem = Trim(textline) 'i.e., clear
  frm_tokens.lst_token_group.AddItem Trim(textline)
 While Trim(textline) <> "N token sets ="
  Line Input #1, textline
 Wend
 Line Input #1, textline
 n_token_sets = Val(textline)
 While Trim(textline) <> "N tokens ="
  Line Input #1, textline
 Wend
 Line Input #1, textline
 token_collection(n_token_groups).n_tokens = Val(textline)
 'While Trim(textline) <> "Token set #"
 ' Line Input #1, textline
 'Wend
 'Line Input #1, textline
 'token_set_num = Val(textline)
 For n = 1 To n_token_sets
 While Trim(textline) <> "Token set #"
  Line Input #1, textline
 Wend
 Line Input #1, textline
 token_set_num = Val(textline)
 token_collection(n_token_groups).add_token_set frm_tokens.lst_token_sets
 token_num = 0
 For i = 1 To token_collection(n_token_groups).n_tokens

   Line Input #1, textline
   token_num = token_num + 1
   token_collection(n_token_groups).set_token token_set_num, token_num, textline
 Next i
```

```
 Next n
 token_num = 0
 Line Input #1, textline
Wend
frm_tokens.lst_token_group.clear
For i = 1 To n_token_groups
frm_tokens.lst_token_group.AddItem token_collection(i).stem
Next i
Line Input #1, textline '### Begining of options ###
Line Input #1, textline
While Trim(textline) <> "### End of options ###"

code = Trim(textline)
Line Input #1, textline
textline = Trim(textline)
Select Case code
Case "mutation_rate"
  mutation_rate = Val(textline)
Case "cross_over_freq"
  cross_over_freq = Val(textline)
Case "frame shift prob"
  frame_shift_prob = Val(textline)
Case "n_runs"
  n_runs = Val(textline)
Case "theta_crit"
  theta_crit = Val(textline)
Case "omega_crit"
  omega_crit = Val(textline)
Case "sigma_crit"
  sigma_crit = Val(textline)
Case "cov_crit"
  cov_crit = Val(textline)
Case "success_criteria"
 success_crit = Val(textline)
Case "pop_size"
   pop_size = Val(textline)
Case "generation_limit"
   generation_limit = Val(textline)
Case "call_method"
   call_method = Trim(textline)
Case "upper fitness limit"
   upper_fitness_limit = Trim(textline)
Case "lower_fitness_limit"
   lower_fitness_limit = Trim(textline)
Case "correlation criteria"
   corr_crit = Trim(textline)

Case "save control"
  save_control = Trim(textline)
Case "save best"
  save_best = Trim(textline)
Case "save output"
   save_output = Trim(textline)
Case "omega block"
omega_block = Trim(textline)
Case "seed type"
```

```
 seed_type = Trim(textline)
Case "seed value"
 seed_value = Trim(textline)
End Select
Line Input #1, textline
Wend
Close #1
'update options

End Sub
'
'
'Sub scale_fitness(fitness() As Single)
''scale by emax, with max fitness at 90% of emax and min fitness at 10% of emax
''first find min and max
'Dim max_fit As Single, min_fit As Single, ef50 As Single, sum_fit As Single
'Dim i As Integer, n As Integer
'Dim emax As Single, emin As Single
'emin = 0.2
'n = UBound(fitness) - LBound(fitness)
'max_fit = -999999
'min_fit = 999999
'For i = LBound(fitness) To UBound(fitness)
'  If fitness(i) > max_fit Then max_fit = fitness(i)
'  If fitness(i) < min_fit Then min_fit = fitness(i)
'  sum_fit = sum_fit + fitness(i)
'Next i
'emax = sum_fit * 2 / n
'ef50 = sum_fit / n
'emin = sum_fit * 0.2 / n
'
'For i = LBound(fitness) To UBound(fitness)
'  fitness(i) = emax * fitness(i) ^ 2 / (ef50 ^ 2 + fitness(i) ^ 2) + emin
'Next i
'MsgBox emax & Chr(9) & ef50 & Chr(9) & emin
'End Sub



'Private Sub map_run()
'Dim n_ind As Integer, this_generation As Integer, max_generation As Integer
'Dim max_values() As Integer
'Dim mapped_values() As Integer
'Dim values() As Integer
'Dim binary() As Boolean
'Dim n_genes As Integer
'Dim n_bits() As Integer
'Dim bin_length As Integer
'n_genes = 5
'max_generation = 6
'n_ind = 3
'ReDim max_values(1 To n_genes)
'ReDim values(1 To n_genes, 1 To n_ind)
'ReDim mapped_values(1 To n_genes, 1 To n_ind)
'max_values(1) = 5
```

```
End Sub
'
Sub map(values() As Integer, max_values() As Integer, mapped_values() As Integer, n_bits() As Integer)
" take unmapped values (1 to max_values) to mapped (0 to 2^ ngenes -1)
" values() is 2 dimensional, (n_genes by n_subject)
" max_valus is 1 dimension (n_genes)
" mapped values is 2 dimensional (n_genes by n_subject)
" n_bits() is 1 dimensino, (n_genes)
Dim i As Integer, n As Integer, p As Integer
" n_bits = 1, max_value = 2;repeat = 0
"  1 -> 0 ;2 -> 1
" n_bits = 2, max_value = 3; repeat = 1
"  1 -> 0 ;2 -> 2:  3 -> 3
" n_bits = 2, max_value = 4; repeat = 0
"  1 -> 0 ;2 -> 1:  3 -> 2; 4 -> 3
" n_bits = 3, max_value = 5; repeat = 3
"  1 -> 0 ;2 -> 2:  3 -> 4; 4 -> 6; 5 -> 7
" n_bits = 3, max_value = 6; repeat = 2
"  1 -> 0 ;2 -> 2:  3 -> 4; 4 -> 5; 5->6;6 -> 7
" n_bits = 3, max_value = 7; repeat = 1
"  1 -> 0 ;2 -> 2:  3 -> 3; 4 -> 4; 5 -> 5; 6 -> 6; 7->7
" n_bits = 3, max_value = 8; repeat = 0
"  1 -> 0 ;2 -> 1:  3 -> 2; 4 -> 3; 5 -> 4; 6 -> 5; 7->6; 8->7
" n_bits = 4, max_value = 9; repeat = 7
"  1 -> 0 ;2 -> 2:  3 -> 4; 4 -> 6; 5 -> 8; 6 -> 10; 7->12; 8->14; 9 -> 15
" n_bits = 4, max_value = 10; repeat = 6
"  1 -> 0 ;2 -> 2:  3 -> 4; 4 -> 6; 5 -> 8; 6 -> 10; 7->12; 8->13;9->14; 10 -> 15
" n_bits = 4, max_value = 11;repeat = 5
"  1 -> 0 ;2 -> 2:  3 -> 4; 4 -> 6; 5 -> 8; 6 -> 10; 7->11; 8->12;9->13; 10 -> 14; 11->15
" n_bits = 4, max_value = 12;repeat = 4
"  1 -> 0 ;2 -> 2:  3 -> 4; 4 -> 6; 5 -> 8; 6 -> 9; 7->10; 8->11;9->12; 10 -> 13; 11->14;12->15
" n_bits = 4, max_value = 13;repeat = 3
"  1 -> 0 ;2 -> 2:  3 -> 4; 4 -> 6; 5 -> 7; 6 -> 8; 7->9; 8->10;9->11;10 -> 12;11->13;12->14;13->15
" n_bits = 4, max_value = 14;repeat = 2
"  1 -> 0 ;2 -> 2:  3 -> 4; 4 -> 5; 5 -> 6; 6 -> 7;7->8; 8->9;9->10;10 -> 11;11->12;12->13;13->14;14->15
Dim repeated As Integer, this_pop As Integer
For this_pop = 1 To UBound(values, 2)
For i = LBound(values, 1) To UBound(values, 1)
repeated = 2 ^ n_bits(i) - max_values(i)
" want 2 * values up to repeated, then 1 * value
If values(i, this_pop) <= repeated + 1 Then
'  mapped_values(i, this_pop) = (values(i, this_pop) - 1) * 2
Else
'  mapped_values(i, this_pop) = (repeated) * 2 + (values(i, this_pop) - repeated - 1)
End If
Next i
Next this_pop
End Sub

Sub unmap(values() As Integer, max_values() As Integer, mapped_values() As Integer, n_bits() As Integer)
' take mapped values (0 to 2^ ngenes -1) back to unmapped (1 to max_values)
' values() is 2 dimensional, (n_genes by n_subject)
' max_valus is 1 dimension (n_genes)
' mapped values is 2 dimensional (n_genes by n_subject)
' n_bits() is 1 dimensino, (n_genes)
Dim i As Integer, repeated As Integer, this_ind As Integer
```

```
start_pos = 1
Next this_ind
End Sub

 Sub mutate(binary() As Boolean, p_mutation As Single)
 Dim this_gene As Integer, this_ind As Integer
 For this_gene = 1 To UBound(binary, 1)
  For this_ind = 1 To UBound(binary, 2)
   If Rnd() < p_mutation Then binary(this_gene, this_ind) = Not (binary(this_gene, this_ind))
  Next this_ind
 Next this_gene
 End Sub

Function InStr_not(string1 As String, string2 As String, string3 As String)
Dim position1 As Long, position2 As Long, start As Long
start = 1
'frm_test.Text1 = string1
'frm_test.Show 1
position1 = InStr(start, string1, string2)
position2 = InStr(start, string1, string3)
While position1 > 0
 If position2 <= position1 And position2 + Len(string3) >= position1 + Len(string2) Then
  start = start + position1
 End If
 position1 = InStr(start, string1, string2)
 position2 = InStr(start, string1, string3)
Wend
InStr_not = position1
End Function


Function from_to(start As Long, string1 As String, from_string As String, to_string As String) As String
'this function return the string that starts with from_string and ends with to_string,
'starting the search at start
Dim new_string As String
Dim start_pos As Long, end_pos As Long
'find the start
start_pos = InStr(start, string1, from_string)
'then the end position
end_pos = InStr(start_pos, string1, to_string)
new_string = Mid(string1, start_pos, end_pos + Len(to_string))

from_to = new_string
End Function
Public Function call_nm(drivename As String, pathname As String, controlfile As String, _
 obj1 As Single, succ1 As Integer, covar As Integer, limit_str As String, check_out As Boolean) As
Double
Dim fitness As Double
Dim theta(1 To max_theta) As Single, setheta(1 To max_theta) As Single
Dim lltheta(1 To max_theta) As Single, ultheta(1 To max_theta) As Single
Dim omega(1 To 30, 1 To 30) As Single, seomega(1 To 30, 1 To 30) As Single
Dim sigma(1 To 30, 1 To 30) As Single, sesigma(1 To 30, 1 To 30) As Single
Dim obj(1 To 2) As Single, rm(1 To 69, 1 To 69) As Single
Dim i As Integer
Dim success(1 To 2) As Single
success(1) = 999
```

```
success(2) = 999
Dim a As Long, p As Integer, ntheta As Integer
Dim out_val As Single
ChDrive drivename
ChDir pathname
If controlfile <> "control" Then FileCopy controlfile, "control"
' need to sent check_out to nmv_exe if false then don't execute
a = nmv_exe(theta(1), lltheta(1), ultheta(1), omega(1, 1), sigma(1, 1), obj(1), success(1), setheta(1), _
    seomega(1, 1), sesigma(1, 1), rm(1, 1))
  fitness = calc_fitness(obj(), success(), setheta(), seomega(), sesigma(), rm(), _
    theta_crit, omega_crit, sigma_crit, cov_crit, ntheta)
run_number = run_number + 1
' hit upper of lower limits?

For i = 1 To ntheta
 If theta(i) <> 0 Then ' check for divide by zero
  If Abs(theta(i) - lltheta(i) / theta(i)) < 0.00000001 Or _
    Abs(theta(i) - ultheta(i) / theta(i)) < 0.000000001 Then
   limit_str = limit_str & Trim(i) & ","
  End If
 End If
Next i
obj1 = obj(1): succ1 = success(1): covar = success(2)
call_nm = fitness
 End Function

 Private Function calc_fitness(obj() As Single, success() As Single, setheta() As Single, _
    seomega() As Single, sesigma() As Single, rm() As Single, _
    theta_crit As Single, omega_crit As Single, sigma_crit As Single, _
    cov_crit As Single, ntheta As Integer) As Single

'return calculated value for fitness, start with obj, subtract theta_crit for each estimated theta etc
Dim i As Integer, neff As Integer, n As Integer
Dim nsigma As Integer
Dim corr_pen As Single ' correlation > 0.95 penalty
Dim cov_pen As Single
'count theta, etas and sigmas
' read from file inputs, created by cfilex (and thetas) in nmtran
Dim iline As String
Dim s_pen As Single
Dim nomega_sets As Integer, nomega As Integer
Dim nsigma_sets As Integer
Dim fixd As Integer, block_num As Integer, nval As Integer
Dim nthfxd As Integer, nomfxd As Integer, nsgfxd As Integer
'MsgBox CurDir
If Dir("INPUTS", vbNormal) <> "" Then
Open "inputs" For Input As #1
Else
calc_fitness = 999999999.99
obj(1) = 999999999.99
Exit Function
End If
Line Input #1, iline
' FIRST LINE IS " NTHETA, NOMEGA, NSIGMA, NTHFXD, NOMFXD, NSGFXD"
Input #1, ntheta, nomega_sets, nsigma_sets, nthfxd, nomfxd, nsgfxd
Line Input #1, iline
```

```
'THIRD LIND IS " NOMBLK , OMDIM, OMFIX"
For i = 1 To nomega_sets
 Input #1, block_num, nval, fixd
  ' get # of etas in block
  If fixd = 0 Then
    For n = 1 To nval
     nomega = nomega + n
    Next n
  End If ' fixd = 0
 Next i

 Line Input #1, iline
For i = 1 To nsigma_sets
 Input #1, block_num, nval, fixd
 nsigma = nsigma + nval * (1 - fixd)
 Next i
 'loop through srm to see if any are larger than 0.95
 neff = ntheta + nomega + nsigma
 corr_pen = 0
 If success(2) = 0 Then ' only do if cov step ran
   For i = 1 To neff
     For n = 1 To i - 1 ' only do lower triangle
      If rm(i, n) > 0.95 Then
        corr_pen = corr_crit
        Exit For
      End If
     Next n
     If corr_pen > 0 Then Exit For
   Next i
 End If
 'now calcuate fitness
If success(2) > 0 Then
cov_pen = cov_crit
Else
cov_pen = 0
End If
If success(1) > 0 Then
s_pen = success_crit
cov_pen = cov_crit
Else
s_pen = 0
End If
calc_fitness = obj(1) + theta_crit * ntheta + omega_crit * nomega _
  + sigma_crit * nsigma + cov_pen + corr_pen + s_pen

Close #1
End Function
Private Sub get_files(files)
files(1) = "nonmem.dll"
files(2) = "freport"
files(3) = "nonmem.lib"
files(4) = "nonmem.exp"
files(5) = "FWARN"
files(6) = "PRDERR"
files(7) = "fsubs"
files(8) = "fsubs.obj"
```

```
mean = sumx / i
If i > 1 Then
sd = Sqr((i * sumxx - (sumx * sumx)) / (i * (i - 1)))
Else
sd = 0
End If
'now draw line from (mean -2d,lower limit) and (mean + 2sd, upper limit)
If sd = 0 Then
 slope = 0
Else
 slope = (upper_fitness_limit - lower_fitness_limit) / (4 * sd)
' y = mx + b
' b =  y - mx
b = 1 - slope * mean
End If
For n = 1 To i
new_fitness(n) = b + slope * fitness(n)
If new_fitness(n) < lower_fitness_limit Then new_fitness(n) = lower_fitness_limit
If new_fitness(n) > upper_fitness_limit Then new_fitness(n) = upper_fitness_limit


Next n
'Open "c:\ga\fitness" For Output As #1
'Dim n_pop As Integer
'For n_pop = 1 To i
'Write #1, temp_fitness(n_pop); new_fitness(n_pop)
'Next n_pop
'Close #1
End Sub

Sub select_pairs(pairs() As Integer, cum_fitness() As Single)
' select individuals based on fitness, put then into pairs
Dim rand As Single, i As Integer, p As Integer, n As Integer, n_ind As Integer
' Open "c:\ga\pairs" For Output As #1
n_ind = UBound(cum_fitness) / 2
For i = 1 To 2
 For n = 1 To n_ind
rand = Rnd()
 p = 1
 While rand > cum_fitness(p)
 p = p + 1
 Wend
 pairs(i, n) = p
' Print #1, rand; cum_fitness(p); p
 Next n
Next i
'Close #1
End Sub
Sub load_data(sheet As vaSpread, dir_name As String)
frm_graphics.CommonDialog1.filename = dir_name & "\*.dat"
Dim Data() As Single, temp As String, varname As String, tmp_num As Integer
Dim data_row() As Single, mdv_col As Integer, n_col As Integer
Dim row_string As String
Dim this_col As Integer, this_row As Integer
Dim next_position As Integer, last_position As Integer
frm_graphics.CommonDialog1.DialogTitle = "NONMEM graphics"
```

```
frm_graphics.CommonDialog1.ShowOpen
If InStr(1, frm_graphics.CommonDialog1.filename, "*") <> 0 Then Exit Sub
If Dir(frm_graphics.CommonDialog1.filename, vbDirectory) = "" Then
MsgBox ("File not found")
Exit Sub
End If

frm_graphics.lst_x_axis.clear
frm_graphics.lst_y_axis.clear
frm_graphics.lst_sort_col.clear
Open frm_graphics.CommonDialog1.filename For Input As #1
'find start of table
While Left(Trim(temp), 12) <> "TABLE NO.  1"
Line Input #1, temp
Wend
frm_graphics.Show
'read headers
temp = Input(1, #1)
frm_graphics.spr_data.row = 0
'read whole line then search for tokens
Line Input #1, row_string
'at least position 11 has to be character
next_position = InStr(11, row_string, " ")
last_position = 1
'find the mdv column
Do While next_position <> 0
   this_col = this_col + 1
   varname = Mid(row_string, last_position, next_position - last_position)
   'if varname is not alpha, there are no variable names
   If IsError(Val(varname)) Then
     MsgBox ("No variable names, next time please use the ""One Header"" option")
     Exit Do
   End If
   If Trim(varname) = "MDV" Then mdv_col = this_col
     frm_graphics.spr_data.col = this_col
     frm_graphics.spr_data.value = Trim(varname)
     frm_graphics.lst_x_axis.AddItem Trim(varname)
     frm_graphics.lst_y_axis.AddItem Trim(varname)
     frm_graphics.lst_sort_col.AddItem Trim(varname)
   last_position = next_position + 1
   'find next space in row
   next_position = InStr(last_position + 11, row_string, " ")
Loop
'then one more
this_col = this_col + 1
If mdv_col = 0 Then
 MsgBox ("No MDV column found, all data will be displayed")
 End If
   varname = Mid(row_string, last_position, Len(row_string) - last_position + 1)
    If Trim(varname) = "MDV" Then mdv_col = this_col
      frm_graphics.spr_data.col = this_col
      frm_graphics.spr_data.value = Trim(varname)
      frm_graphics.lst_x_axis.AddItem Trim(varname)
      frm_graphics.lst_y_axis.AddItem Trim(varname)
      frm_graphics.lst_sort_col.AddItem Trim(varname)
n_col = this_col
```

```
ReDim data_row(1 To n_col)
'read data'
While EOF(1) = False
 Input #1, row_string
 For this_col = 1 To n_col
  data_row(this_col) = Val(Mid(row_string, 1 + (this_col - 1) * 12, 12))
 Next this_col
 If mdv_col <> 0 Then ' can we check if mdv = 1?
  If data_row(mdv_col) = 0 Then
     this_row = this_row + 1
     ' use data
      frm_graphics.spr_data.row = this_row
     For this_col = 1 To n_col
       frm_graphics.spr_data.col = this_col
       frm_graphics.spr_data.value = data_row(this_col)
     Next this_col
  End If
 Else
  ' use it regardless if mdv not present
    this_row = this_row + 1
     frm_graphics.spr_data.row = this_row
     For this_col = 1 To n_col
       frm_graphics.spr_data.col = this_col
       frm_graphics.spr_data.value = data_row(this_col)
     Next this_col
 End If
Wend
Close #1
frm_graphics.spr_data.MaxRows = this_row
'frm_graphics_interface.Show
   End Sub

Sub scan_tokens()
'look for (), (1-0), unmatched (), unmatched { }, {({}), ({}), ({})}
'only if they occur before a ";" in the token
Dim short_token As String, token As String, i As Integer, ok As Boolean
Dim this_token_group As Integer, this_token_set As Integer, this_token As Integer
Dim pos1 As Integer, pos2 As Integer
ok = True
For this_token_group = 1 To n_token_groups
 For this_token_set = 1 To token_collection(this_token_group).n_token_sets
    For this_token = 1 To token_collection(this_token_group).n_tokens
   token = token_collection(this_token_group).get_token(this_token_set, this_token)
   ' first get the part left of ";"
   If InStr(token, ";") = 0 Then
   short_token = token
   Else
   short_token = Left(token, InStr(1, token, ";") - 1)
   End If
   ' look for THETA(1-0)x
   For i = 0 To 9
     If InStr(UCase(short_token), "ETA(" & Trim(str(i)) & ")") <> 0 Then
       MsgBox ("Number " & str(i) & " in token = " & token & " stem = " &
token_collection(this_token_group).stem & _
         " Token set # " & this_token_set & " Token # " & this_token)
      ok = False
```

```
      End If
      If InStr(UCase(short_token), "EPS(" & Trim(str(i)) & ")") <> 0 Then
       MsgBox ("Number " & str(i) & " in token = " & token & " stem = " &
token_collection(this_token_group).stem & _
       " Token set # " & this_token_set & " Token # " & this_token)
       ok = False
      End If
    Next i
    'check for unbalance () have to loop through until all ( are found
    'before next (
    pos1 = 1 'position of first (
    pos2 = 1 'position of next (
    While InStr(pos1 + 1, short_token, "(") <> 0
     pos1 = InStr(pos1 + 1, short_token, "(")
     If pos1 > 0 Then
       If InStr(pos1, short_token, ")") = 0 Then
       MsgBox ("Unmatched ( in " & token & " stem = " & token_collection(this_token_group).stem & _
           " Token set # " & this_token_set & " Token # " & this_token)
       ok = False
       End If
      End If
    Wend
     Next this_token
  Next this_token_set
Next this_token_group
If ok = True Then MsgBox "No errors found"
End Sub

Public Function make_int(this_ind As Integer) As Double
'need to start with binary (genome) not values
Dim this_digit As Integer, length_genome As Integer, rval As Double
length_genome = UBound(genome, 1)
For this_digit = 1 To length_genome
If genome(this_digit, this_ind) = True Then rval = rval + 2 ^ (this_digit - 1)
Next this_digit
make_int = rval
End Function
Public Sub clear_form(Form As vaSpread)
Dim this_row As Integer, this_col As Integer
With Form
For this_row = 1 To n_runs
.row = this_row
  For this_col = 1 To 8
  .col = this_col
  .text = ""
  Next this_col
Next this_row
End With
End Sub


Public Sub get_stats(directory As String, obj As Single, fitness As Single, covar As Boolean, success As
Boolean)
'read the files  parms and return the statistics, send to calc_fitness
ChDir directory
```

```
Close #1
If Dir("PARMS", vbNormal) <> "" Then
Open "PARMS" For Input As #1
End If
Close #1
End Sub
Public Function check_token(box As TextBox) As Boolean
check_token = True
Dim temp_str As String
Dim n_parens As Integer
Dim pos As Integer
temp_str = box.text
pos = InStr(1, temp_str, " ")
While pos <> 0
temp_str = Left(temp_str, pos - 1) & _
        Right(temp_str, Len(temp_str) - pos)
pos = InStr(1, temp_str, " ")
Wend
box.text = temp_str
pos = InStr(1, temp_str, "(")
While pos <> 0
n_parens = n_parens + 1
temp_str = Left(temp_str, pos - 1) & _
        Right(temp_str, Len(temp_str) - pos)
pos = InStr(1, temp_str, "(")
Wend
' now subtract one for each ")"
pos = InStr(1, temp_str, ")")
While pos <> 0
n_parens = n_parens - 1
temp_str = Left(temp_str, pos - 1) & _
        Right(temp_str, Len(temp_str) - pos)
pos = InStr(1, temp_str, ")")
Wend
If n_parens > 0 Then
  MsgBox "Too many ""("""s"
  box.SelStart = InStr(1, box.text, "(") - 1
  box.SelLength = 1
  check_token = False
  Exit Function
End If
If n_parens < 0 Then
  MsgBox "Too many "")"""s"
  box.SelStart = InStr(1, box.text, ")") - 1
  box.SelLength = 1
  check_token = False
  Exit Function
End If
' and now the {}
temp_str = box.text
pos = InStr(1, temp_str, "{")
While pos <> 0
n_parens = n_parens + 1
temp_str = Left(temp_str, pos - 1) & _
        Right(temp_str, Len(temp_str) - pos)
pos = InStr(1, temp_str, "{")
```

```
Wend
' now subtract one for each ")"
pos = InStr(1, temp_str, "}")
While pos <> 0
n_parens = n_parens - 1
temp_str = Left(temp_str, pos - 1) & _
        Right(temp_str, Len(temp_str) - pos)
pos = InStr(1, temp_str, "}")
Wend
If n_parens > 0 Then
  MsgBox "Too many ""{""s"
  box.SelStart = InStr(1, box.text, "{") - 1
  box.SelLength = 1
  check_token = False
  Exit Function
End If
If n_parens < 0 Then
  MsgBox "Too many ""}""s"
  box.SelStart = InStr(1, box.text, "}") - 1
  box.SelLength = 1
  check_token = False
  Exit Function
End If
End Function


Public Sub load_results(spread As vaSpread, chart As MSChart)
' recurse through the directories, calculate the fitness for each run and write to spread sheet
Dim i As Integer
Dim n_ind As Integer, this_ind As Integer
Dim cur_gen_dir As String, cur_ind_dir As String, this_row As Integer
Dim obj(1 To 2) As Single, success(1 To 2) As Single, covar As Boolean
Dim fitness() As Double, scaled_fitness() As Single, temp_fitness() As Single
Dim n_gen As Integer, max_ind As Integer, max_gen As Integer, max_x As Single
' how many individuals
this_gen = 1: this_ind = 1: max_gen = 0: max_ind = 0
this_row = 0
'cur_gen_dir = home_directory & "\" & Trim(str(this_gen))
'cur_ind_dir = cur_gen_dir & "\" & Trim(str(this_ind))
'For this_gen = 1 To last_gen
'While Dir(cur_gen_dir, vbDirectory) = Trim(str(this_gen))
'   If this_gen > max_gen Then max_gen = this_gen
' While Dir(cur_ind_dir, vbDirectory) = Trim(str(this_ind))
'   If this_ind > max_ind Then max_ind = this_ind
'   this_ind = this_ind + 1
'    cur_ind_dir = cur_gen_dir & "\" & Trim(str(this_ind))
' Wend 'cur_ind_dir
  this_ind = 1

'cur_gen_dir = home_directory & "\" & Trim(str(this_gen))
'Next this_gen
'this_gen = last_gen
'Wend 'cur_gen_dir
ReDim fitness(1 To pop_size)
ReDim scaled_fitness(1 To pop_size): ReDim temp_fitness(1 To pop_size)
' initialize plot axis for generations
```

```
frm_main.spr_result.MaxRows = pop_size * generation_limit
max_x = generation_limit
initialize_plot generation_limit
this_gen = 1: this_ind = 1
For this_gen = 1 To last_gen
 cur_gen_dir = home_directory & "\" & Trim(str(this_gen))
'While Dir(cur_gen_dir, vbDirectory) = Trim(str(this_gen))
 'While Dir(cur_ind_dir, vbDirectory) = Trim(str(this_ind))
  For this_ind = 1 To pop_size
 cur_ind_dir = cur_gen_dir & "\" & Trim(str(this_ind))
  'we need obj success, covar, fitness, boundary for theta.
  'the calc scaled fitness
  'read input, parms
  read_results cur_ind_dir, obj, success, covar, fitness(this_ind)
  this_row = this_row + 1
  With frm_main.spr_result
    .row = this_row
    .col = 1: .text = obj(1)
    .col = 2: If success(1) = 0 Then .text = "Yes" Else .text = "No"
    .col = 3: If success(2) = 0 Then .text = "Yes" Else .text = "No"
    .col = 4: .text = fitness(this_ind)
    .col = 8:    .text = this_gen
    .col = 9:    .text = this_ind
   End With
  temp_fitness(this_ind) = fitness(this_ind)
'  this_ind = this_ind + 1
 'cur_ind_dir = cur_gen_dir & "\" & Trim(str(this_ind))
  Next this_ind 'end of individual while

scale_fitness scaled_fitness(), temp_fitness()
'update plot
update_plot temp_fitness(), scaled_fitness()
 'this_gen = this_gen + 1
 'cur_gen_dir = home_directory & "\" & Trim(str(this_gen))
 'this_ind = 1
 'cur_ind_dir = cur_gen_dir & "\" & Trim(str(this_ind))
Next this_gen
run_number = last_gen * pop_size
frm_main.pgb_gen = last_gen
frm_main.pgb_gen.max = generation_limit
frm_main.pgb_ind.max = pop_size

frm_main.pgb_ind = 1

'Wend 'end of generation while
'MsgBox frm_main.spr_result.MaxRows
End Sub


Sub read_results(this_dir As String, ByRef obj() As Single, ByRef success() As Single, covar As Boolean,
fitness As Double)
Dim theta(1 To max_theta) As Single, setheta(1 To max_theta) As Single
Dim ntheta As Integer, nomega As Integer, nsigma As Integer, ntheta_fixed As Integer, nomega_fixed As
Integer, nsigma_fixed As Integer
Dim lltheta(1 To max_theta) As Single, ultheta(1 To max_theta) As Single
Dim omega(1 To 30, 1 To 30) As Single, seomega(1 To 30, 1 To 30) As Single
```

```
Exit Sub
End If
Close 1
End Sub


Function count_max_omega() As Integer
' go to the tokens and find how many unique omegas are there.
' count the number of unique ETA(??) where ?? is A - AZ
Dim used_eta(1 To 52) As Boolean
Dim this_eta As Integer, token_string As String, check_string As String
Dim this_token_group As Integer, this_token_set
Dim n_omega As Integer, n_token_sets As Integer
Dim this_token As Integer, n_tokens As Integer
Dim test_string As String, n_sets As Integer
Dim control_string As String, n_token_omegas ' number of omegas in tokens (ie., "A")
control_string = UCase(frm_main.txt_code)
' well assume there are less than 10 omegas in the control file and less than 27 on the token sets
test_string = "ETA(" & Trim(Chr(49)) & ")"
' change all the THETAs to xxx
control_string = sub_string(control_string, "THETA", "XXXXX")
While InStr(1, control_string, test_string) <> 0 And n_omega < 10
 n_omega = n_omega + 1
 test_string = "ETA(" * Trim(Chr(49 + n_omega)) & ")"
Wend
For this_token_group = 1 To n_token_groups
 n_token_sets = token_collection(this_token_group).n_token_sets
 For this_token_set = 1 To n_token_sets
  ' loop through set to see if it is used
  n_tokens = token_collection(this_token_group).n_tokens
   For this_token = 1 To n_tokens
    token_string = token_string & vbCrLf & token_collection(this_token_group).get_token(this_token_set,
this_token)
   Next this_token
 Next this_token_set
Next this_token_group
' get rid of THETA (to XXXX)
token_string = sub_string(token_string, "THETA", "XXXXX")
 ' frm_text.txt_text = token_string
 ' frm_text.Show 1, frm_main
 test_string = "{ETA(" & Trim(Chr(65)) & ")}"
 While InStr(1, token_string, test_string) <> 0
 n_token_omegas = n_token_omegas + 1
 test_string = "{ETA(" & Trim(Chr(64 + n_token_omegas)) & ")}"
Wend
count_max_omega = n_omega + n_token_omegas
End Function

Function count_etan(control As String) As Integer
Dim control_string As String, test_string As String
Dim test_n As Integer
control_string = UCase(control)
' well assume there are less than 10 omegas in the control file and less than 27 on the token sets
test_n = 1
test_string = "ETA(" & Trim(str(test_n)) & ")"
' change all the THETAs to xxx
```

```
control_string = sub_string(control_string, "THETA", "XXXXX")
While InStr(1, control_string, test_string) <> 0 And n_omega < 10
  test_n = test_n + 1
  test_string = "ETA(" & Trim(str(test_n)) & ")"
Wend
count_etan = test_n - 1
End Function
```

File token_group.cls

```
VERSION 1.0 CLASS
BEGIN
  MultiUse = -1  'True
END
Attribute VB_Name = "token_group"
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = True
Attribute VB_PredeclaredId = False
Attribute VB_Exposed = False
Attribute VB_Ext_KEY = "SavedWithClassBuilder" ,"Yes"
Attribute VB_Ext_KEY = "Top_Level" ,"Yes"
'local variable(s) to hold property value(s)
Private local_stem As String 'local copy
Private local_n_tokens As Integer ' number of tokens in set , e.g.,
theta(next),(0,1,100) = 2 tokens
Private local_n_token_sets As Integer
Private token_sets(1 To 50, 1 To 10) As String
Option Explicit

Public Sub remove_token_set(ByVal position As Integer)
Dim row As Integer, col As Integer
For row = position To local_n_token_sets - 1
 For col = 1 To n_tokens
 token_sets(row, col) = token_sets(row + 1, col)
 Next col
Next row
local_n_token_sets = local_n_token_sets - 1
End Sub

'Private all_sets As Collection
Public Sub add_token_set(lst_sets As ListBox)
local_n_token_sets = local_n_token_sets + 1
get_token_set lst_sets
End Sub
Public Property Get stem() As String
    stem = local_stem
End Property
Public Property Let stem(ByVal lstem As String)
    local_stem = lstem
End Property

Public Property Get n_tokens() As Integer
    n_tokens = local_n_tokens
End Property
Public Property Get n_token_sets() As Integer
    n_token_sets = local_n_token_sets
End Property
Public Property Let n_tokens(n As Integer)
    local_n_tokens = n
End Property

Public Sub get_token_set(this_list As ListBox)
Dim i As Integer, n As Integer
```

```
Dim tok_str As String
this_list.clear
For i = 1 To local_n_token_sets
tok_str = i & "  "
  For n = 1 To local_n_tokens
  tok_str = tok_str & "(" & token_sets(i, n) & ") "
  Next n
this_list.AddItem tok_str
Next

End Sub
Public Sub get_tokens(this_list As ListBox, this_token_set As Integer)
Dim i As Integer
this_list.clear
If this_token_set <> 0 Then
For i = 1 To local_n_tokens
   this_list.AddItem "(" & token_sets(this_token_set, i) & ") "
  Next i
End If
End Sub

Public Sub get_tokens_lines(this_list As ListBox, this_token_set As
Integer)
Dim i As Integer
Dim str As String
Dim start As Integer, last As Integer
this_list.clear
For i = 1 To local_n_tokens
  str = token_sets(this_token_set, i)
  While InStr(1, str, "{crlf}") <> 0
  start = InStr(1, str, "{crlf}")
  last = Len(str) - InStr(1, str, "{crlf}") + Len("{crlf}")
     str = Trim(Left(str, start) & vbCrLf & _
           Trim(Right(str, last)))
   Wend
  this_list.AddItem "(" & str & ") "
Next i
End Sub
Public Function get_token(ByVal set_num As Integer, ByVal token_num As
Integer) As String
get_token = token_sets(set_num, token_num)
End Function
Public Function get_token_with_lines(ByVal set_num As Integer, ByVal
token_num As Integer) As String
Dim str As String
Dim start As Integer, last As Integer
  str = token_sets(set_num, token_num)
  While InStr(1, str, "{crlf}") <> 0
  start = InStr(1, str, "{crlf}") - 1
  last = Len(str) - InStr(1, str, "{crlf}") - Len("{crlf}") + 1
     str = Trim(Left(str, start) & vbCrLf & _
           Trim(Right(str, last)))
   Wend

get_token_with_lines = str
End Function
```

```
Public Sub set_token(ByVal set_num As Integer, ByVal token_num As
Integer, value As String)
' we need to replace crlf by another character - {crlf}
Dim start As Integer, last As Integer
While InStr(1, value, vbCrLf) <> 0
start = InStr(1, value, vbCrLf) - 1
last = Len(value) - start - 2
value = Trim(Left(value, start)) & "{crlf}" & _
        Trim(Right(value, last))
Wend
token_sets(set_num, token_num) = value

End Sub

Public Sub clear()
Dim i As Integer, n As Integer
For i = 1 To n_token_sets
  For n = 1 To n_tokens
  token_sets(i, n) = ""
  Next n
Next i
local_n_token_sets = 0
local_n_tokens = 0
local_stem = ""
End Sub
```

```
   Width           =   855
End
Begin TabDlg.SSTab SSTab1
   Height          =   8295
   Left            =   360
   TabIndex        =   0
   Top             =   0
   Width           =   12255
   _ExtentX        =   21616
   _ExtentY        =   14631
   _Version        =   393216
   TabOrientation  =   3
   Tab             =   2
   TabHeight       =   520
   BeginProperty Font {0BE35203-8F91-11CE-9DE3-00AA004BB851}
      Name            =   "Arial"
      Size            =   11.25
      Charset         =   0
      Weight          =   400
      Underline       =   0   'False
      Italic          =   0   'False
      Strikethrough   =   0   'False
   EndProperty
   TabCaption(0)   =   "Control"
   TabPicture(0)   =   "frm_main.frx":0442
   Tab(0).ControlEnabled=   0   'False
   Tab(0).Control(0)=   "txt_code"
   Tab(0).ControlCount=   1
   TabCaption(1)   =   "Result Plot"
   TabPicture(1)   =   "frm_main.frx":045E
   Tab(1).ControlEnabled=   0   'False
   Tab(1).Control(0)=   "MSChart1"
   Tab(1).ControlCount=   1
   TabCaption(2)   =   "Results table"
   TabPicture(2)   =   "frm_main.frx":047A
   Tab(2).ControlEnabled=   -1  'True
   Tab(2).Control(0)=   "spr_result"
   Tab(2).Control(0).Enabled=   0   'False
   Tab(2).ControlCount=   1
   Begin VB.TextBox txt_code
      Height          =   7815
      Left            =   -74160
      MultiLine       =   -1  'True
      ScrollBars      =   2  'Vertical
      TabIndex        =   7
      Top             =   240
      Width           =   9495
   End
   Begin MSChartLib.MSChart MSChart1
      Height          =   7815
      Left            =   -74880
      OleObjectBlob   =   "frm_main.frx":0496
      TabIndex        =   1
      Top             =   120
      Width           =   10815
   End
   Begin FPSpread.vaSpread spr_result
```

```
      Height          =   7935
      Left            =   240
      TabIndex        =   8
      Top             =   120
      Width           =   11115
      _Version        =   131077
      _ExtentX        =   19606
      _ExtentY        =   13996
      _StockProps     =   64
      BeginProperty Font {0BE35203-8F91-11CE-9DE3-00AA004BB851}
         Name            =   "MS Sans Serif"
         Size            =   8.25
         Charset         =   0
         Weight          =   700
         Underline       =   0   'False
         Italic          =   0   'False
         Strikethrough   =   0   'False
      EndProperty
      MaxCols         =   11
      ScrollBars      =   2
      ScrollBarShowMax=   0   'False
      SpreadDesigner  =   "frm_main.frx":252A
      UserResize      =   2
      VisibleCols     =   500
      VisibleRows     =   500
   End
End
Begin ComctlLib.ProgressBar pgb_ind
   Height          =   210
   Left            =   1440
   TabIndex        =   2
   Top             =   8400
   Width           =   5175
   _ExtentX        =   9128
   _ExtentY        =   370
   _Version        =   327682
   Appearance      =   1
End
Begin ComctlLib.ProgressBar pgb_gen
   Height          =   210
   Left            =   1440
   TabIndex        =   4
   Top             =   8760
   Width           =   5175
   _ExtentX        =   9128
   _ExtentY        =   370
   _Version        =   327682
   Appearance      =   1
End
Begin VB.Label Label3
   Caption         =   "Unique models"
   Height          =   255
   Left            =   10080
   TabIndex        =   13
   Top             =   8640
   Width           =   1335
End
```

```
Begin VB.Label lbl_count
   BackColor        =   &H80000009&
   BorderStyle      =   1  'Fixed Single
   Caption          =   "0"
   Height           =   375
   Left             =   11640
   TabIndex         =   12
   Top              =   8520
   Width            =   855
End
Begin VB.Label Label1
   Caption          =   "Individuals"
   Height           =   255
   Left             =   360
   TabIndex         =   6
   Top              =   8400
   Width            =   975
End
Begin VB.Label Label2
   Caption          =   "Generations"
   Height           =   255
   Left             =   360
   TabIndex         =   5
   Top              =   8760
   Width            =   1095
End
Begin VB.Menu file
   Caption          =   "File"
   WindowList       =   -1  'True
   Begin VB.Menu new
      Caption          =   "&New"
   End
   Begin VB.Menu open
      Caption          =   "&Open"
   End
   Begin VB.Menu Save
      Caption          =   "&Save"
   End
   Begin VB.Menu Load
      Caption          =   "Load results"
   End
   Begin VB.Menu save_as
      Caption          =   "S&ave As"
   End
   Begin VB.Menu Exit
      Caption          =   "E&xit"
   End
   Begin VB.Menu break
      Caption          =   "-"
   End
   Begin VB.Menu files
      Caption          =   "Files"
      Index            =   1
      Visible          =   0   'False
   End
   Begin VB.Menu files
      Caption          =   "Files"
```

```
      Index           =   2
      Visible         =   0   'False
   End
   Begin VB.Menu files
      Caption         =   "Files"
      Index           =   3
      Visible         =   0   'False
   End
   Begin VB.Menu files
      Caption         =   "Files"
      Index           =   4
      Visible         =   0   'False
   End
End
Begin VB.Menu edit
   Caption         =   "Edit"
   Begin VB.Menu Edit_token_set
      Caption         =   "Edit Token Set"
   End
   Begin VB.Menu sort
      Caption         =   "Sort Results"
   End
   Begin VB.Menu print
      Caption         =   "Print"
   End
   Begin VB.Menu copy
      Caption         =   "Copy"
   End
End
Begin VB.Menu Run
   Caption         =   "Run"
   Begin VB.Menu check_out
      Caption         =   "Check Out"
   End
   Begin VB.Menu ga_Run
      Caption         =   "GA Run"
   End
   Begin VB.Menu continue_run
      Caption         =   "Continue GA run"
   End
   Begin VB.Menu full_grid
      Caption         =   "Full Grid Search"
   End
   Begin VB.Menu debug
      Caption         =   "Debug"
   End
End
Begin VB.Menu option
   Caption         =   "Options"
   Begin VB.Menu settings
      Caption         =   "Settings"
   End
End
Begin VB.Menu help
   Caption         =   "Help"
End
End
```

```
Attribute VB_Name = "frm_main"
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = False
Attribute VB_PredeclaredId = True
Attribute VB_Exposed = False
Private cur_model_file_name As String
Private file_name As String ' just the file name without the path
Option Explicit
Private Sub but_stop_run_Click()
stop_run = True
End Sub

Private Sub copy_Click()
If SSTab1.Tab = 1 Then
MSChart1.EditCopy
MsgBox "result plot chart copied to clipboard"
End If
If SSTab1.Tab = 2 Then
spr_result.col = -1
spr_result.row = -1
spr_result.Action = 22
End If
End Sub
Private Sub debug_Click()
frm_debug.Show 1, Me
End Sub



Private Sub files_Click(Index As Integer)

cur_model_file_name = start_files(Index)
get_model cur_model_file_name
Dim pos As Integer
pos = 1
While InStr(pos + 1, cur_model_file_name, "\") > 0
pos = InStr(pos + 1, cur_model_file_name, "\")
Wend
home_directory = Left(cur_model_file_name, pos - 1)
ChDir (home_directory)
file_name = Right(cur_model_file_name, Len(cur_model_file_name) - pos)
home_drive = Left(home_directory, 2)
ChDrive (home_drive)
End Sub

Private Sub Form_Unload(Cancel As Integer)
End
End Sub

Private Sub Load_Click()

If MsgBox("Load results from " & home_directory & " ?", vbOKCancel) <>
vbOK Then Exit Sub
load_results frm_main.spr_result, frm_main.MSChart1

End Sub
```

```
Private Sub opt_pause_Click()
opt_pause.Enabled = False
paused = True
Opt_resume.Enabled = True
End Sub
'
Private Sub opt_pause_DblClick()
opt_pause.Enabled = False
paused = True
Opt_resume.Enabled = True
End Sub

Private Sub Opt_resume_Click()
Opt_resume.Enabled = False
paused = False
opt_pause.Enabled = True
End Sub
'
Private Sub Opt_resume_DblClick()
Opt_resume.Enabled = False
paused = False
opt_pause.Enabled = True
End Sub

Private Sub print_click()
If SSTab1.Tab = 1 Then
End If
If SSTab1.Tab = 2 Then
spr_result.col = -1
spr_result.row = -1
spr_result.Action = 22
End If

End Sub
Private Sub New_Click()
Me.txt_code.text = ""
set_default_options
End Sub
Private Sub set_default_options()

End Sub


Private Sub sort_Click()
frm_sort_results.Show
End Sub

Private Sub spr_result_Click(ByVal col As Long, ByVal row As Long)
Dim gen As Integer, ind As Integer
Dim text As String, textline As String
Dim file_name As String
Select Case col
 Case 5
 spr_result.col = 8
 spr_result.row = row
 If spr_result.value = "" Then
 MsgBox "No results available"
```

```
Exit Sub
End If
If save_output = False Then
MsgBox "Output file not saved, see options"
Exit Sub
End If
gen = spr_result.value
spr_result.col = 9
ind = spr_result.value
If spr_result.value = "" Then
MsgBox "No results available"
Exit Sub
End If
file_name = home_directory & "\" & gen & "\" & ind & "\output"
If Dir(file_name, vbNormal) = "" Then
MsgBox "Output file not found"
Exit Sub
End If

Open file_name For Input As #1
Do While Not EOF(1) ' Loop until end of file.
   Line Input #1, textline ' Read line into variable.
   text = text & textline & vbCrLf
Loop
frm_text.Caption = "Output file"
frm_text.txt_text = text
Me.Hide
frm_text.Show
Close #1    ' Close file.
Case 6
spr_result.col = 8
spr_result.row = row
If spr_result.value = "" Then
MsgBox "No results available"
Exit Sub
End If
If save_control = False Then
MsgBox "control file not saved, see options"
Exit Sub
End If
gen = spr_result.value
spr_result.col = 9
If spr_result.value = "" Then
MsgBox "No results available"
Exit Sub
End If
ind = spr_result.value
file_name = home_directory & "\" & gen & "\" & ind & "\control"
Open file_name For Input As #1
Do While Not EOF(1) ' Loop until end of file.
   Line Input #1, textline ' Read line into variable.
   text = text & textline & vbCrLf
Loop
frm_text.Caption = "Control file"
frm_text.txt_text = text
frm_text.Show 1, frm_main
Close #1    ' Close file.
```

```
Case 7

spr_result.col = 8
spr_result.row = row
If spr_result.value = "" Then
MsgBox "Generation not available"
Exit Sub
End If
gen = spr_result.value
spr_result.col = 9
ind = spr_result.value
file_name = home_directory & "\" & gen & "\" & ind
load_data frm_graphics.spr_data, file_name

frm_graphics.Show
End Select
End Sub


Private Sub check_out_Click()
Dim n_runs As Integer
n_runs = frm_options.txt_pop_size * frm_options.txt_generations
frm_main.spr_result.MaxRows = n_runs
SSTab1.Tab = 2
stop_run = False
frm_main.but_stop_run.Enabled = True
ga_runner True, True
frm_main.but_stop_run.Enabled = False
End Sub

Private Sub continue_run_Click()
stop_run = False
ga_runner False, False
End Sub
Private Sub Edit_token_set_Click()
Me.Hide
frm_tokens.Show

End Sub
Private Sub exit_Click()

Dim i As Integer
SaveSetting appname:="NM_GA", section:="Startup", _
            Key:="N", setting:=n_files
For i = 1 To n_files
SaveSetting appname:="NM_GA", section:="Startup", _
            Key:="File" & str(i), setting:=start_files(i)
Next i
'SaveSetting appname:="NM_GA", section:="Startup", _
'            Key:="File" & str(1), setting:="c:\570\amy\ga\570b.mdl"
'
'SaveSetting appname:="NM_GA", section:="Startup", _
'            Key:="File" & str(2), setting:="c:\570\amy\ga\570c.mdl"

End
End Sub
Private Sub Form_Load()
```

```
'ChDir "c:\ga\"
'cur_model_file_name = "c:\ga\gen.mdl"
'get_model cur_model_file_name
'frm_tokens.lst_token_group.ListIndex = 0
'frm_tokens.lst_token_sets.ListIndex = 0

End Sub

Private Sub full_grid_Click()
stop_run = False
grid_search
End Sub
Private Sub ga_Run_Click()
Dim n_runs As Integer
n_runs = frm_options.txt_pop_size * frm_options.txt_generations
frm_main.spr_result.MaxRows = n_runs
SSTab1.Tab = 2
stop_run = False
frm_main.but_stop_run.Enabled = True
ga_runner True, False
frm_main.but_stop_run.Enabled = False
'frm_inter_results.Hide

End Sub
Private Sub open_Click()
Me.CommonDialog1.DialogTitle = "Open GA model"
ChDir (home_directory)
Me.CommonDialog1.InitDir = home_directory
Me.CommonDialog1.filename = "*.mdl"
Me.CommonDialog1.ShowOpen
If Me.CommonDialog1.filename = "*.dat" Or Me.CommonDialog1.filename =
"" Or Me.CommonDialog1.filename = "*.mdl" Then
Exit Sub
End If
cur_model_file_name = Me.CommonDialog1.filename
get_model Me.CommonDialog1.filename
frm_tokens.lst_token_group.ListIndex = 0
frm_tokens.lst_token_sets.ListIndex = 0
' get home directory name
Dim pos As Integer
pos = 1
While InStr(pos + 1, cur_model_file_name, "\") > 0
pos = InStr(pos + 1, cur_model_file_name, "\")
Wend
home_directory = Left(cur_model_file_name, pos - 1)
file_name = Right(cur_model_file_name, Len(cur_model_file_name) - pos)
ChDir (home_directory)
home_drive = Left(home_directory, 2)
ChDrive (home_drive)


End Sub

Private Sub save_as_Click()
Dim file_name As String
Me.CommonDialog1.DialogTitle = "Save GA model file"
Me.CommonDialog1.InitDir = home_directory
```

```
Me.CommonDialog1.Filter = "*.mdl"
Me.CommonDialog1.filename = "*.mdl"
Me.CommonDialog1.ShowSave
file_name = Me.CommonDialog1.filename
If Trim(file_name) = "" Then Exit Sub
If file_name = "" Then
Exit Sub
Else
' set home directory
Dim pos As Integer
pos = 1
While InStr(pos + 1, file_name, "\") > 0
pos = InStr(pos + 1, file_name, "\")
Wend
home_directory = Left(file_name, pos - 1)
ChDir (home_directory)
home_drive = Left(home_directory, 2)
ChDrive (home_drive)
file_name = Right(file_name, Len(file_name) - pos)
ChDrive (home_drive)
cur_model_file_name = file_name
save_model (file_name)
Dim i As Integer, n As Integer
For i = 1 To n_files
  If start_files(i) = home_drive & "\" & home_directory & "\" &
file_name Then
' remove it
    For n = i To n_files - 1 Step 1
       start_files(n) = start_files(n + 1)
    Next n
  n_files = n_files - 1
    start_files(n_files + 1) = ""
    Exit For
  End If
Next i
If n_files < 4 Then n_files = n_files + 1
For i = n_files To 2 Step -1
start_files(i) = start_files(i - 1)
frm_main.files(i).Caption = start_files(i)
Next i
start_files(1) = home_drive & "\" & home_directory & "\" & file_name
frm_main.files(1).Caption = start_files(1)
End If
End Sub

Private Sub Save_Click()
save_model (file_name)
End Sub

Private Sub settings_Click()
Me.Hide
set_options
frm_options.Show 1, Me
End Sub

Private Sub SSTab1_Click(PreviousTab As Integer)
```

```
If SSTab1.Tab = 2 Then
sort.Enabled = True
Else
sort.Enabled = False
End If
End Sub
```

```
      TabIndex        =   0
      Top             =   3240
      Width           =   975
   End
End
Attribute VB_Name = "frm_debug"
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = False
Attribute VB_PredeclaredId = True
Attribute VB_Exposed = False
Option Explicit

Private Sub but_scan_numbs_Click()
scan_tokens
End Sub

Private Sub Command1_Click()
Me.Hide
End Sub
```

File frm_edit_token.frm

```
VERSION 5.00
Object = "{F9043C88-F6F2-101A-A3C9-08002B2F49FB}#1.1#0"; "Comdlg32.ocx"
Begin VB.Form frm_edit_token
   Caption         =   "Edit Token"
   ClientHeight    =   5355
   ClientLeft      =   3900
   ClientTop       =   3645
   ClientWidth     =   7065
   LinkTopic       =   "Form1"
   ScaleHeight     =   5355
   ScaleWidth      =   7065
   Begin MSComDlg.CommonDialog CommonDialog1
      Left            =   480
      Top             =   4680
      _ExtentX        =   847
      _ExtentY        =   847
      _Version        =   327680
   End
   Begin VB.TextBox txt_token
      BeginProperty Font
         Name            =   "MS Sans Serif"
         Size            =   12
         Charset         =   0
         Weight          =   400
         Underline       =   0   'False
         Italic          =   0   'False
         Strikethrough   =   0   'False
      EndProperty
      Height          =   3615
      HideSelection   =   0   'False
      Left            =   960
      MultiLine       =   -1  'True
      ScrollBars      =   2  'Vertical
      TabIndex        =   2
      Top             =   480
      Width           =   5655
   End
   Begin VB.CommandButton but_cancel
      Caption         =   "Cancel"
      Height          =   495
      Left            =   4200
      TabIndex        =   1
      Top             =   4320
      Width           =   1095
   End
   Begin VB.CommandButton but_done
      Caption         =   "Done"
      Height          =   495
      Left            =   2040
      TabIndex        =   0
      Top             =   4320
      Width           =   1095
   End
   Begin VB.Label Label1
      Caption         =   "Token"
```

```
      Height          =   375
      Left            =   120
      TabIndex        =   3
      Top             =   960
      Width           =   855
   End
   Begin VB.Menu file
      Caption         =   "File"
      Begin VB.Menu import
         Caption         =   "Import"
      End
      Begin VB.Menu export
         Caption         =   "Export"
      End
      Begin VB.Menu save
         Caption         =   "Save and close"
      End
      Begin VB.Menu exit
         Caption         =   "Exit (don't save)"
      End
   End
End
Attribute VB_Name = "frm_edit_token"
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = False
Attribute VB_PredeclaredId = True
Attribute VB_Exposed = False

Private Sub but_done_Click()
If check_token(txt_token) = False Then
Exit Sub
End If
Me.Hide
frm_tokens.Show
End Sub

Private Sub import_Click()
Dim code As String, textline As String
Me.CommonDialog1.DialogTitle = "Import token"
Me.CommonDialog1.filename = "*.txt"
Me.CommonDialog1.ShowOpen
If Me.CommonDialog1.filename = "*.txt" Or _
   Me.CommonDialog1.filename = "" Then
   Exit Sub
End If
   Open Me.CommonDialog1.filename For Input As #1
  Do While Not EOF(1) ' Loop until end of file.
    Line Input #1, textline ' Read line into variable.
    Debug.Print textline    ' Print to Debug window.
    code = code & textline & vbCrLf
Loop
Close #1
Me.txt_token = code
End Sub


Private Sub export_Click()
```

```
Dim code As String, textline As String
Dim new_code As String
Me.CommonDialog1.DialogTitle = "Export token"
Me.CommonDialog1.filename = "*.txt"
Me.CommonDialog1.ShowSave
If Me.CommonDialog1.filename = "*.txt" Or _
   Me.CommonDialog1.filename = "" Then
   Exit Sub
End If
   Open Me.CommonDialog1.filename For Output As #1
   code = Me.txt_token
Print #1, code  ' Print text to file.
Close #1
End Sub
```

File frm_graphics.frm

```
VERSION 5.00
Object = "{B02F3647-766B-11CE-AF28-C3A2FBE76A13}#2.5#0"; "SS32X25.OCX"
Object = "{BDC217C8-ED16-11CD-956C-0000C04E4C0A}#1.1#0"; "TABCTL32.OCX"
Object = "{827E9F53-96A4-11CF-823E-000021570103}#1.0#0"; "GRAPHS32.OCX"
Object = "{F9043C88-F6F2-101A-A3C9-08002B2F49FB}#1.1#0"; "Comdlg32.ocx"
Begin VB.Form frm_graphics
   Caption         =   "Graphics"
   ClientHeight    =   8835
   ClientLeft      =   60
   ClientTop       =   630
   ClientWidth     =   10695
   LinkTopic       =   "Form1"
   ScaleHeight     =   8835
   ScaleWidth      =   10695
   Begin MSComDlg.CommonDialog CommonDialog1
      Left            =   3240
      Top             =   8400
      _ExtentX        =   847
      _ExtentY        =   847
      _Version        =   327680
   End
   Begin TabDlg.SSTab SSTab1
      Height          =   8535
      Left            =   120
      TabIndex        =   0
      Top             =   120
      Width           =   10110
      _ExtentX        =   17833
      _ExtentY        =   15055
      _Version        =   393216
      TabOrientation  =   1
      Tabs            =   1
      TabsPerRow      =   10
      TabHeight       =   520
      TabCaption(0)   =   "Main"
      TabPicture(0)   =   "frm_graphics.frx":0000
      Tab(0).ControlEnabled=   -1  'True
      Tab(0).Control(0)=   "Label2"
      Tab(0).Control(0).Enabled=   0   'False
      Tab(0).Control(1)=   "Label1"
      Tab(0).Control(1).Enabled=   0   'False
      Tab(0).Control(2)=   "spr_data"
      Tab(0).Control(2).Enabled=   0   'False
      Tab(0).Control(3)=   "lst_sort_col"
      Tab(0).Control(3).Enabled=   0   'False
      Tab(0).Control(4)=   "lst_y_axis"
      Tab(0).Control(4).Enabled=   0   'False
      Tab(0).Control(5)=   "lst_x_axis"
      Tab(0).Control(5).Enabled=   0   'False
      Tab(0).Control(6)=   "but_histos"
      Tab(0).Control(6).Enabled=   0   'False
      Tab(0).Control(7)=   "But_done"
      Tab(0).Control(7).Enabled=   0   'False
      Tab(0).Control(8)=   "but_make_plot"
      Tab(0).Control(8).Enabled=   0   'False
```

```
   Width           =   495
   _Version        =   327680
   _ExtentX        =   873
   _ExtentY        =   450
   _StockProps     =   96
   GraphStyle      =   2
   GraphType       =   9
   LeftTitleStyle  =   1
   RandomData      =   0
   SymbolData      =   "13~13~7~13"
   SymbolSize      =   10
End
Begin VB.CheckBox chk_ind_sorted_plots
   Caption         =   "Individual Sorted Plots"
   Height          =   255
   Left            =   7080
   TabIndex        =   15
   Top             =   5870
   Width           =   2175
End
Begin VB.CheckBox chk_ind_y_plots
   Caption         =   "Individual Y Plots"
   Height          =   255
   Left            =   3720
   TabIndex        =   14
   Top             =   5870
   Width           =   1815
End
Begin VB.CheckBox chk_plot_matrix
   Caption         =   "Plot matrix"
   Height          =   255
   Left            =   3960
   TabIndex        =   13
   Top             =   6720
   Width           =   1095
End
Begin VB.CheckBox chk_sort_col
   Caption         =   "Use Sort Item"
   Height          =   255
   Left            =   7080
   TabIndex        =   9
   Top             =   3480
   Width           =   1455
End
Begin VB.CheckBox chk_abs_value
   Caption         =   "Use Absolute Value"
   Height          =   255
   Left            =   3720
   TabIndex        =   8
   Top             =   6240
   Width           =   1935
End
Begin VB.CheckBox chk_unit_line
   Caption         =   "Unit Line"
   Height          =   255
   Left            =   600
   TabIndex        =   7
```

```
      _Version        =   131077
      _ExtentX        =   16748
      _ExtentY        =   5318
      _StockProps     =   64
      BeginProperty Font {0BE35203-8F91-11CE-9DE3-00AA004BB851}
         Name            =   "MS Sans Serif"
         Size            =   8.25
         Charset         =   0
         Weight          =   700
         Underline       =   0   'False
         Italic          =   0   'False
         Strikethrough   =   0   'False
      EndProperty
      SpreadDesigner  =   "frm_graphics.frx":001C
   End
   Begin VB.Label Label1
      Caption         =   "X axis"
      Height          =   255
      Left            =   480
      TabIndex        =   12
      Top             =   3360
      Width           =   1575
   End
   Begin VB.Label Label2
      Caption         =   "Y axis"
      Height          =   255
      Left            =   3720
      TabIndex        =   11
      Top             =   3360
      Width           =   1575
   End
End
Begin VB.Menu file
   Caption         =   "File"
   Begin VB.Menu Close
      Caption         =   "Close"
   End
End
Begin VB.Menu edit
   Caption         =   "Edit"
   Begin VB.Menu copy
      Caption         =   "&Copy"
      Shortcut        =   ^C
   End
End
End
Attribute VB_Name = "frm_graphics"
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = False
Attribute VB_PredeclaredId = True
Attribute VB_Exposed = False
Option Explicit

Private Sub but_done_Click()
Unload frm_graphics
frm_main.Show
End Sub
```

```
Private Sub but_histos_Click()
Dim i As Integer
With frm_histo.lst_histo
  .clear
For i = 1 To lst_x_axis.ListCount
   .AddItem lst_x_axis.list(i - 1)
Next i
End With
Me.Hide
frm_histo.Show
End Sub

Private Sub but_make_plot_Click()
Dim i As Integer, n_tabs As Integer, n_plots As Integer
If lst_x_axis.ListIndex < 0 Then
 MsgBox "Please select one or more x variables"
 Exit Sub
End If
If lst_y_axis.ListIndex < 0 Then
 MsgBox "Please select one or more y variables"
 Exit Sub
End If
If chk_sort_col.value = 1 And lst_sort_col.ListIndex < 0 Then
 MsgBox "Please select a sort variable"
 Exit Sub
End If
' figure out how may x and y selected
Dim n_x As Integer, n_y As Integer, xs(1 To 20) As Integer, ys(1 To 20)
As Integer

For i = 0 To lst_x_axis.ListCount - 1
 If lst_x_axis.Selected(i) = True Then
   n_x = n_x + 1
   xs(n_x) = i
 End If
Next i
For i = 0 To lst_y_axis.ListCount - 1
 If lst_y_axis.Selected(i) = True Then
   n_y = n_y + 1
   ys(n_y) = i
  End If
Next i
n_tabs = SSTab1.Tabs
n_plots = Graph.count - 1 ' index starts at 0, but we don't use 0
' single x, single y, single plot
If n_x = 1 And n_y = 1 And chk_plot_matrix.value = 0 And chk_sort_col =
0 Then
  n_tabs = n_tabs + 1
  n_plots = n_plots + 1
  SSTab1.Tabs = n_tabs
  SSTab1.Tab = n_tabs - 1
  SSTab1.TabCaption(n_tabs - 1) = lst_x_axis.list(lst_x_axis.ListIndex)
& "/" & lst_y_axis.list(lst_y_axis.ListIndex)
  Load Graph(n_plots)
  make_xy lst_x_axis.ListIndex + 1, lst_y_axis.ListIndex + 1,
Graph(n_plots)
```

```
End If
' single x, several
If n_x = 1 And n_y > 1 And chk_plot_matrix.value = 0 And chk_sort_col =
0 Then
' one plot, one x, many y
make_multiy xs(1), ys, n_y
End If
If n_x > 1 And chk_plot_matrix.value = 0 And chk_sort_col = 0 Then
make_multi_x xs, ys, n_x, n_y
End If
' plot matrix, one plot
If chk_plot_matrix.value = 1 And chk_sort_col = 0 Then
  plot_matrix xs, ys, n_x, n_y
End If
' sorted, one x, one y
If chk_sort_col = 1 And n_x = 1 And n_y = 1 Then
make_sorted_xy xs(1), ys(1), Me.lst_sort_col.ListIndex
End If

End Sub

Private Sub make_sorted_xy(X As Integer, y As Integer, sort_col As
Integer)
' NOTE THAT LISTINDICES START AT 0
Dim n_subs As Integer, max_obs As Integer, this_point As Integer
Dim this_sub As Integer, this_graph_point As Integer, this_tab As
Integer
Dim n_data As Integer, i As Integer, this_plot As Integer, this_id As
Integer
' first need to pass through data, and count max obs per subject
n_subs = 1
spr_data.col = sort_col + 1
spr_data.row = 1
this_sub = spr_data.value
For i = 2 To spr_data.MaxRows
 spr_data.row = i
 If spr_data.value <> this_sub Then
  n_subs = n_subs + 1
  this_sub = spr_data.value
 End If
Next i
' add a tab
SSTab1.Tabs = SSTab1.Tabs + 1
this_tab = SSTab1.Tabs
SSTab1.Tab = this_tab - 1
this_plot = Graph.count - 1
this_plot = this_plot + 1
Load Graph(this_plot)
With Graph(this_plot)
   .Visible = True
   .Enabled = True
   .Top = 400
   .Left = 400
   .width = 9200
   .height = 7400
  ' how many data
  n_data = spr_data.MaxRows
```

```
If n_y > n_x Then max_dim = n_y
width = (SSTab1.width - left_margin * 2) / max_dim - (max_dim - 1) *
gap
' 400 FOR TAB ROW
' need to adjust height for # of rows of tabs
height = (SSTab1.height - top_margin * 2 - 400) / max_dim - (max_dim -
1) * gap
n_plots = n_x * n_y + start_plot
 n_data = spr_data.MaxRows
While this_plot < n_plots
 this_tab = this_tab + 1
 SSTab1.Tabs = this_tab
 SSTab1.Tab = this_tab - 1
 SSTab1.TabCaption(this_tab - 1) = "Matrix"
   For i = 1 To n_x
     If this_plot = n_plots Then Exit For
     For n = 1 To n_y
     If this_plot = n_plots Then Exit For
     this_plot = this_plot + 1
      Load Graph(this_plot)
      With Graph(this_plot)
      .Visible = True
      .BorderStyle = 0
       .Left = left_margin + (n - 1) * width
       .width = width
       .Top = top_margin + (i - 1) * height
       .height = height
       .BottomTitle = lst_x_axis.list(xs(i))
       .LeftTitle = lst_y_axis.list(ys(n))
       .Enabled = True
        .NumSets = 1
         .NumPoints = n_data
        For p = 1 To n_data
        Dim junk As String
        spr_data.row = p
'        spr_data.col = 0
'        junk = spr_data.text
'        spr_data.col = 1
'        junk = spr_data.text
        spr_data.col = xs(i) + 1
         .XPos(p) = Val(spr_data.text)
        spr_data.col = ys(n) + 1
         .Data(p) = Val(spr_data.text)
        Next p
         .SymbolData = 13 ' solid cirle
         .SymbolSize = 20 + 18 * max_dim

 If Me.chk_trend_line.value = 1 Then
  If opt_line.value = True Then .LineStats = 8
  If opt_smooth.value = True Then
  .CurveOrder = 2
  .LineStats = 16
  .PatternedLines = 1
  .PatternData = 1
  End If
 End If
       .DrawMode = graphDraw
```

```
End With

SSTab1.TabCaption(this_tab - 1) = spr_data.text
End Sub
Private Sub make_xy(X As Integer, y As Integer, this_graph As Graph)
Dim n_data As Integer, i As Integer
Dim sumx As Double, sumxx As Double, sumy As Double, sumxy As Double,
sumyy As Double
'Dim maxx As Single, minx As Single
Dim slope As Single, intercept As Single, xval As Single, yval As
Single
'SSTab1.TabCaption(SSTab1.Tabs - 1) = lst_x_axis.list(X) & "/" &
lst_y_axis.list(Y)
With this_graph
  .Visible = True
  .Enabled = True
  .Top = 400
  .Left = 400
  .width = 9200
  .height = 7400
 ' how many data
 n_data = spr_data.MaxRows
  .NumSets = 1
  .NumPoints = n_data
'  maxx = -999999999
'  minx = 999999999
 For i = 1 To n_data
 spr_data.row = i
 spr_data.col = X
  xval = Val(spr_data.text)
  .XPos(i) = xval
  sumx = sumx + xval
  sumxx = sumxx + xval * xval
 ' If xval > maxx Then maxx = xval
 ' If xval < minx Then minx = xval
  spr_data.col = y
  yval = Val(spr_data.text)
  .Data(i) = yval
  sumy = sumy + yval
  sumyy = sumyy + yval * yval
  sumxy = sumxy + yval * xval
 Next i
  .SymbolData = 13 ' solid cirle
  .SymbolSize = 30
 spr_data.col = y
 spr_data.row = 0
  .LeftTitle = spr_data.text
 spr_data.col = X
  .BottomTitle = spr_data.text
' add trend line
 If Me.chk_trend_line.value = 1 Then
  If opt_line.value = True Then .LineStats = 8
  If opt_smooth.value = True Then
  .CurveOrder = 2
  .LineStats = 16
  .PatternedLines = 1
  .PatternData = 1
```

```
 End If
End If
Dim rsquare As Double
Dim denom As Double
denom = Sqr((n_data * sumxx - sumx * sumx) * (n_data * sumyy - sumy *
sumy))
If denom > 0.00000000001 Then
rsquare = (n_data * sumxy - sumx * sumy) / denom
Else
rsquare = 1
End If
  .BottomTitle = .BottomTitle & "  R^2 = " & Format(rsquare, "0.000")
  .DrawMode = graphDraw
End With
End Sub


Private Sub chk_sort_col_Click()
If chk_sort_col.value = 1 Then
lst_sort_col.Enabled = True
Else
lst_sort_col.Enabled = False
End If
End Sub

Private Sub chk_trend_line_Click()
If chk_trend_line.value = 1 Then
opt_line.Enabled = True
opt_smooth.Enabled = True
Else

opt_line.Enabled = False
opt_smooth.Enabled = False
End If
End Sub

Private Sub copy_Click()
MsgBox "Nothing to copy"
End Sub
```

File frm_histo.frm

```
VERSION 5.00
Object = "{BDC217C8-ED16-11CD-956C-0000C04E4C0A}#1.1#0"; "TABCTL32.OCX"
Object = "{827E9F53-96A4-11CF-823E-000021570103}#1.0#0"; "GRAPHS32.OCX"
Begin VB.Form frm_histo
   Caption         =   "Make Histograms"
   ClientHeight    =   9045
   ClientLeft      =   60
   ClientTop       =   630
   ClientWidth     =   12120
   LinkTopic       =   "Form1"
   ScaleHeight     =   9045
   ScaleWidth      =   12120
   StartUpPosition =   3  'Windows Default
   Begin VB.TextBox txt_nbins
      Height          =   285
      Left            =   1320
      TabIndex        =   14
      Text            =   "10"
      Top             =   6480
      Width           =   855
   End
   Begin VB.CheckBox chk_autobins
      Caption         =   "Auto select bins"
      Height          =   375
      Left            =   480
      TabIndex        =   12
      Top             =   5880
      Width           =   1695
   End
   Begin VB.Frame Frame4
      Height          =   1095
      Left            =   360
      TabIndex        =   9
      Top             =   4560
      Width           =   1695
      Begin VB.OptionButton opt_lin
         Caption         =   "Linear scale"
         Height          =   255
         Left            =   240
         TabIndex        =   11
         Top             =   240
         Value           =   -1  'True
         Width           =   1215
      End
      Begin VB.OptionButton opt_log
         Caption         =   "log scale"
         Height          =   255
         Left            =   240
         TabIndex        =   10
         Top             =   600
         Width           =   1095
      End
   End
   Begin VB.TextBox txt_n_rows
      Height          =   405
```

```
      Top             =   0
      Width           =   9495
      _ExtentX        =   16748
      _ExtentY        =   15478
      _Version        =   327681
      TabOrientation  =   1
      Tabs            =   1
      TabsPerRow      =   5
      TabHeight       =   520
      TabPicture(0)   =   "frm_histo.frx":0000
      Tab(0).ControlEnabled=   -1  'True
      Tab(0).Control(0)=   "Graph(0)"
      Tab(0).Control(0).Enabled=   0   'False
      Tab(0).ControlCount=   1
      Begin GraphsLib.Graph Graph
         Height          =   375
         Index           =   0
         Left            =   480
         TabIndex        =   3
         Top             =   480
         Visible         =   0   'False
         Width           =   615
         _Version        =   327680
         _ExtentX        =   1085
         _ExtentY        =   661
         _StockProps     =   96
         BorderStyle     =   1
         GraphType       =   3
         RandomData      =   0
      End
   End
   Begin VB.Label Label2
      Caption         =   "n bins"
      Height          =   255
      Left            =   480
      TabIndex        =   13
      Top             =   6480
      Width           =   735
   End
   Begin VB.Menu file
      Caption         =   "File"
      Begin VB.Menu exit
         Caption         =   "Exit"
      End
   End
   Begin VB.Menu edit
      Caption         =   "Edit"
      Begin VB.Menu copy
         Caption         =   "&Copy"
         Shortcut        =   ^C
      End
   End
End
Attribute VB_Name = "frm_histo"
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = False
Attribute VB_PredeclaredId = True
```

```
Attribute VB_Exposed = False
 Private cur_plot As Integer
Private Sub but_make_plot_Click()
Dim i As Integer, n_tabs As Integer, n_plots As Integer
If lst_histo.ListIndex < 0 Then
 MsgBox "Please select one or more x variables"
 Exit Sub
End If
' figure out how may x and y selected
Dim n_x As Integer, xs(1 To 20) As Integer

For i = 0 To lst_histo.ListCount - 1
If lst_histo.Selected(i) = True Then
n_x = n_x + 1
xs(n_x) = i
End If
Next i
n_tabs = tab_histo.Tabs
' only add a tab if this is not the first
n_plots = Graph.Count - 1 ' index starts at 0, but we don't use 0
 If n_plots = 0 Then n_tabs = 0
If n_x = 1 Then
 n_tabs = n_tabs + 1
n_plots = n_plots + 1
tab_histo.Tabs = n_tabs
tab_histo.Tab = n_tabs - 1
tab_histo.TabCaption(n_tabs - 1) = lst_histo.list(lst_histo.ListIndex)
Load Graph(n_plots)
With Graph(n_plots)
   .Visible = True
   .Enabled = True
   .Top = 400
   .Left = 400
   .width = 9200
   .height = 7400
  ' how many data
 n_data = frm_graphics.spr_data.MaxRows
   .NumSets = 1
   .NumPoints = n_data
make_histo lst_histo.ListIndex + 1, Graph(n_plots)

End With
End If

End Sub

Sub make_histo(X As Integer, this_graph As Graph)

End Sub

Private Sub chk_autobins_Click()
If chk_autobins.value = 0 Then
txt_nbins.Enabled = True
Else
txt_nbins.Enabled = False
End If
End Sub
```

```
Private Sub copy_Click()
If cur_plot = 0 Then
MsgBox "Please select a plot"
Else
Graph(cur_plot).DrawMode = graphCopy
End If
End Sub

Private Sub exit_Click()
frm_graphics.Show
Me.Hide
End Sub

Private Sub Form_Terminate()
frm_graphics.Show
End Sub

Private Sub Graph_Click(Index As Integer)
cur_plot = Index
End Sub
```

```
   Tab(1).ControlEnabled=   0   'False
   Tab(1).Control(0)=   "spr_result"
   Tab(1).ControlCount=   1
   Begin MSChartLib.MSChart MSChart1
      Height          =   5655
      Left            =   1200
      OleObjectBlob   =   "frm_intermediate_results.frx":0038
      TabIndex        =   2
      Top             =   120
      Width           =   10335
   End
   Begin FPSpread.vaSpread spr_result
      Height          =   5895
      Left            =   -74880
      TabIndex        =   3
      Top             =   120
      Width           =   11520
      _Version        =   131077
      _ExtentX        =   20320
      _ExtentY        =   10398
      _StockProps     =   64
      BeginProperty Font {0BE35203-8F91-11CE-9DE3-00AA004BB851}
         Name            =   "MS Sans Serif"
         Size            =   8.25
         Charset         =   0
         Weight          =   700
         Underline       =   0   'False
         Italic          =   0   'False
         Strikethrough   =   0   'False
      EndProperty
      MaxCols         =   14
      ScrollBars      =   2
      ScrollBarShowMax=   0   'False
      SpreadDesigner  =   "frm_intermediate_results.frx":20C4
      UserResize      =   2
      VisibleCols     =   500
      VisibleRows     =   500
   End
End
Begin VB.CommandButton but_stop_run
   Caption         =   "Stop Run"
   Height          =   375
   Left            =   9840
   TabIndex        =   0
   Top             =   6600
   Width           =   855
End
Begin ComctlLib.ProgressBar pgb_gen
   Height          =   210
   Left            =   2280
   TabIndex        =   5
   Top             =   6840
   Width           =   5175
   _ExtentX        =   9128
   _ExtentY        =   370
   _Version        =   327682
   Appearance      =   1
```

File frm_new_group.frm

```
VERSION 5.00
Begin VB.Form frm_new_group
   Caption         =   "New Token Group"
   ClientHeight    =   3195
   ClientLeft      =   5220
   ClientTop       =   3735
   ClientWidth     =   4680
   LinkTopic       =   "Form1"
   ScaleHeight     =   3195
   ScaleWidth      =   4680
   Begin VB.TextBox txt_stem
      Height          =   375
      Left            =   1440
      TabIndex        =   4
      Top             =   720
      Width           =   1455
   End
   Begin VB.TextBox txt_n_tokens
      Height          =   375
      Left            =   1440
      TabIndex        =   2
      Text            =   "1"
      Top             =   1200
      Width           =   1455
   End
   Begin VB.CommandButton but_cancel
      Caption         =   "Cancel"
      Height          =   495
      Left            =   2400
      TabIndex        =   1
      Top             =   2400
      Width           =   1335
   End
   Begin VB.CommandButton but_done
      Caption         =   "Done"
      Height          =   495
      Left            =   720
      TabIndex        =   0
      Top             =   2400
      Width           =   1335
   End
   Begin VB.Label Label2
      Caption         =   "Stem "
      Height          =   375
      Left            =   240
      TabIndex        =   5
      Top             =   720
      Width           =   975
   End
   Begin VB.Label Label1
      Caption         =   "# of Tokens"
      Height          =   375
      Left            =   240
      TabIndex        =   3
      Top             =   1200
```

```
      Width           =   975
   End
End
Attribute VB_Name = "frm_new_group"
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = False
Attribute VB_PredeclaredId = True
Attribute VB_Exposed = False

Private Sub but_cancel_Click()
Me.txt_n_tokens = -999
Me.txt_stem = -999
Me.Hide
frm_tokens.Show
End Sub

Private Sub but_done_Click()
Me.Hide
frm_tokens.Show
End Sub
```

File frm_options.frm

```
VERSION 5.00
Begin VB.Form frm_options
   Caption         =   "Form1"
   ClientHeight    =   7485
   ClientLeft      =   5265
   ClientTop       =   3360
   ClientWidth     =   8625
   LinkTopic       =   "Form1"
   ScaleHeight     =   7485
   ScaleWidth      =   8625
   Begin VB.CheckBox chk_save_best
      Caption         =   "Save best?"
      Height          =   255
      Left            =   4800
      TabIndex        =   43
      Top             =   3960
      Value           =   1  'Checked
      Width           =   3135
   End
   Begin VB.Frame Frame2
      Caption         =   "Random seed"
      Height          =   1455
      Left            =   4680
      TabIndex        =   38
      Top             =   4440
      Width           =   3495
      Begin VB.TextBox txt_rnd_seed
         Enabled         =   0   'False
         Height          =   375
         Left            =   2040
         TabIndex        =   42
         Text            =   "1"
         Top             =   840
         Width           =   615
      End
      Begin VB.OptionButton opt_rnd_user
         Caption         =   "User Defined"
         Height          =   255
         Left            =   240
         TabIndex        =   41
         Top             =   960
         Width           =   1335
      End
      Begin VB.OptionButton opt_rnd_default
         Caption         =   "Use Default"
         Height          =   255
         Left            =   240
         TabIndex        =   40
         Top             =   240
         Value           =   -1  'True
         Width           =   1335
      End
      Begin VB.OptionButton opt_rnd_clock
         Caption         =   "Use Clock"
         Height          =   255
```

```
      Left            =   240
      TabIndex        =   39
      Top             =   600
      Width           =   1335
   End
End
Begin VB.CheckBox chk_non_diag_omega
   Caption         =   "Include ga for non diagonal OMEGA"
   Height          =   375
   Left            =   4800
   TabIndex        =   37
   Top             =   3480
   Value           =   1  'Checked
   Width           =   2895
End
Begin VB.TextBox txt_frame_shift_prob
   Height          =   285
   Left            =   2760
   TabIndex        =   35
   Text            =   "0.01"
   Top             =   1320
   Width           =   1455
End
Begin VB.CheckBox chk_save_output
   Caption         =   "Save output file"
   Height          =   375
   Left            =   4800
   TabIndex        =   34
   Top             =   3000
   Value           =   1  'Checked
   Width           =   2775
End
Begin VB.CheckBox chk_save_control
   Caption         =   "Save control file"
   Height          =   375
   Left            =   4800
   TabIndex        =   33
   Top             =   2640
   Value           =   1  'Checked
   Width           =   2775
End
Begin VB.TextBox txt_generations
   Height          =   285
   Left            =   2760
   TabIndex        =   31
   Text            =   "20"
   Top             =   6120
   Width           =   1455
End
Begin VB.TextBox txt_succ_crit
   Height          =   285
   Left            =   2760
   TabIndex        =   29
   Text            =   "0.3"
   Top             =   5160
   Width           =   1455
End
```

```
Begin VB.TextBox txt_corr_crit
   Height          =   285
   Left            =   2760
   TabIndex        =   27
   Text            =   "50"
   Top             =   3720
   Width           =   1455
End
Begin VB.TextBox txt_lower_limit
   Height          =   285
   Left            =   2760
   TabIndex        =   24
   Text            =   "0.3"
   Top             =   4680
   Width           =   1455
End
Begin VB.TextBox txt_upper_limit
   Height          =   285
   Left            =   2760
   TabIndex        =   23
   Text            =   "2"
   Top             =   4200
   Width           =   1455
End
Begin VB.TextBox txt_cov_crit
   Height          =   285
   Left            =   2760
   TabIndex        =   21
   Text            =   "1000"
   Top             =   3240
   Width           =   1455
End
Begin VB.Frame Frame1
   Caption         =   "NONMEM call"
   Height          =   1095
   Left            =   4440
   TabIndex        =   18
   Top             =   360
   Width           =   2535
   Begin VB.OptionButton opt_dll
      Caption         =   "DLL (NT only)"
      Height          =   255
      Left            =   120
      TabIndex        =   20
      Top             =   240
      Width           =   1455
   End
   Begin VB.OptionButton opt_exe
      Caption         =   "EXE (NT or 9?)"
      Height          =   255
      Left            =   120
      TabIndex        =   19
      Top             =   600
      Value           =   -1  'True
      Width           =   1935
   End
End
```

```
Begin VB.TextBox txt_cross_over_freq
   Height          =   285
   Left            =   2760
   TabIndex        =   4
   Text            =   "0.8"
   Top             =   360
   Width           =   1455
End
Begin VB.TextBox txt_mutation_rate
   Height          =   285
   Left            =   2760
   TabIndex        =   2
   Text            =   "0.001"
   Top             =   840
   Width           =   1455
End
Begin VB.CommandButton but_cancel
   Caption         =   "Cancel"
   Height          =   495
   Left            =   4680
   TabIndex        =   1
   Top             =   6720
   Width           =   1095
End
Begin VB.CommandButton but_done
   Caption         =   "Done"
   Height          =   495
   Left            =   2280
   TabIndex        =   0
   Top             =   6720
   Width           =   1095
End
Begin VB.Label Label8
   Caption         =   "Frame Shift Probability"
   Height          =   255
   Left            =   240
   TabIndex        =   36
   Top             =   1320
   Width           =   1695
End
Begin VB.Label Label14
   Caption         =   "Generation limit"
   Height          =   255
   Left            =   240
   TabIndex        =   32
   Top             =   6120
   Width           =   1455
End
Begin VB.Label Label13
   Caption         =   "Success Criteria"
   Height          =   255
   Left            =   240
   TabIndex        =   30
   Top             =   5160
   Width           =   2055
End
Begin VB.Label Label12
```

```
   Caption         =   "Penalty for corr > 0.95"
   Height          =   255
   Left            =   240
   TabIndex        =   28
   Top             =   3720
   Width           =   2055
End
Begin VB.Label Label11
   Caption         =   "Lower limit of scaled fitness"
   Height          =   255
   Left            =   240
   TabIndex        =   26
   Top             =   4680
   Width           =   2055
End
Begin VB.Label Label10
   Caption         =   "Upper limit of scaled fitness"
   Height          =   255
   Left            =   240
   TabIndex        =   25
   Top             =   4200
   Width           =   2295
End
Begin VB.Label Label9
   Caption         =   "Covariance criteria"
   Height          =   255
   Left            =   240
   TabIndex        =   22
   Top             =   3240
   Width           =   1335
End
Begin VB.Label Label7
   Caption         =   "Population size"
   Height          =   255
   Left            =   240
   TabIndex        =   17
   Top             =   5640
   Width           =   1095
End
Begin VB.Label Label6
   Caption         =   "Number of threads"
   Height          =   255
   Left            =   4920
   TabIndex        =   15
   Top             =   1800
   Width           =   1695
End
Begin VB.Label Label5
   Caption         =   "Sigma criteria"
   Height          =   255
   Left            =   240
   TabIndex        =   11
   Top             =   2760
   Width           =   1335
End
Begin VB.Label Label4
   Caption         =   "Omega criteria"
```

```
      Height          =   255
      Left            =   240
      TabIndex        =   9
      Top             =   2280
      Width           =   1335
   End
   Begin VB.Label Label3
      Caption         =   "Theta Criteria"
      Height          =   255
      Left            =   240
      TabIndex        =   8
      Top             =   1800
      Width           =   1695
   End
   Begin VB.Label Label2
      Caption         =   "Cross over Frequency"
      Height          =   255
      Left            =   240
      TabIndex        =   5
      Top             =   240
      Width           =   1695
   End
   Begin VB.Label Label1
      Caption         =   "Mutation rate"
      Height          =   255
      Left            =   240
      TabIndex        =   3
      Top             =   840
      Width           =   975
   End
End
Attribute VB_Name = "frm_options"
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = False
Attribute VB_PredeclaredId = True
Attribute VB_Exposed = False

Private Sub but_cancel_Click()
Me.Hide
frm_main.Show
End Sub

Private Sub but_done_Click()
mutation_rate = Me.txt_mutation_rate
cross_over_freq = Me.txt_cross_over_freq
frame_shift_prob = Me.txt_frame_shift_prob
theta_crit = Me.txt_theta_crit
omega_crit = Me.txt_omega_crit
sigma_crit = Me.txt_sigma_crit
cov_crit = Me.txt_cov_crit
success_crit = Me.txt_succ_crit
generation_limit = Me.txt_generations
lower_fitness_limit = Me.txt_lower_limit
upper_fitness_limit = Me.txt_upper_limit
seed_value = Me.txt_rnd_seed.text
If Me.opt_rnd_clock = True Then seed_type = "clock"
If Me.opt_rnd_default = True Then seed_type = "default"
```

```
If Me.opt_rnd_user = True Then seed_type = "user"
corr_crit = Me.txt_corr_crit
If opt_dll = True Then call_method = "dll"
If opt_exe = True Then call_method = "exe"
If chk_save_control = 1 Then
save_control = True
Else
save_control = False
End If
If chk_save_best = 1 Then
save_best = True
Else
save_best = False
End If
If chk_save_output = 1 Then
save_output = True
Else
save_output = False
End If
pop_size = Me.txt_pop_size
If pop_size Mod 2 <> 0 Then
MsgBox "Population size must be even number"
Me.txt_pop_size.SelStart = 0
Me.txt_pop_size.SelLength = Len(Me.txt_pop_size)
Me.txt_pop_size.SetFocus
Exit Sub
End If
' need to redimension genome for non diagonal omega
Dim g2dim As Integer
If Me.chk_non_diag_omega = 1 Then
 omega_block = True
 Else
 omega_block = False
 End If
Me.Hide
frm_main.Show
End Sub

Private Sub chk_non_diag_omega_Click()
If chk_non_diag_omega = True Then
omega_block = True
Else
omega_block = False
End If
End Sub

Private Sub Form_Load()
set_options
End Sub
Private Sub opt_both_limit_Click()
txt_time.Enabled = True
txt_generations.Enabled = True
End Sub

Private Sub opt_generations_Click()
  txt_time.Enabled = False
txt_generations.Enabled = True
```

```
End Sub
Private Sub opt_time_Click()
  txt_time.Enabled = True
txt_generations.Enabled = False
End Sub


Private Sub opt_rnd_clock_Click()
seed_type = "clock"
Me.txt_rnd_seed.Enabled = False
End Sub

Private Sub opt_rnd_default_Click()
seed_type = "default"
Me.txt_rnd_seed.Enabled = False
End Sub

Private Sub opt_rnd_user_Click()
seed_type = "user"
Me.txt_rnd_seed.Enabled = True

End Sub

Private Sub txt_rnd_seed_lostfocus()
On Error GoTo num_error
seed_value = Me.txt_rnd_seed

Exit Sub
num_error:
MsgBox ("Please enter a number")
Me.txt_rnd_seed.SetFocus
Me.txt_rnd_seed.SelStart = 0
Me.txt_rnd_seed.SelLength = Len(Me.txt_rnd_seed.text)

On Error Resume Next
End Sub
```

```
Me.Hide
End Sub
Private Sub spr_result_Click(ByVal Col As Long, ByVal Row As Long)
If Col > 4 Then MsgBox "col = " & Col & "  row = " & Row
End Sub
```

File frm_sort_results.frm

```
VERSION 5.00
Begin VB.Form frm_sort_results
   Caption         =   "Sort Results"
   ClientHeight    =   3975
   ClientLeft      =   60
   ClientTop       =   345
   ClientWidth     =   6930
   LinkTopic       =   "Form1"
   ScaleHeight     =   3975
   ScaleWidth      =   6930
   StartUpPosition =   3  'Windows Default
   Begin VB.CommandButton but_cancel
      Caption         =   "Cancel"
      Height          =   495
      Left            =   3720
      TabIndex        =   7
      Top             =   3240
      Width           =   1095
   End
   Begin VB.CommandButton but_Sort
      Caption         =   "Sort"
      Height          =   495
      Left            =   1920
      TabIndex        =   6
      Top             =   3240
      Width           =   1095
   End
   Begin VB.ListBox lst_third_sort
      Height          =   1230
      ItemData        =   "frm_sort_results.frx":0000
      Left            =   4800
      List            =   "frm_sort_results.frx":0016
      TabIndex        =   4
      Top             =   1320
      Width           =   1335
   End
   Begin VB.ListBox lst_second_sort
      Height          =   1230
      ItemData        =   "frm_sort_results.frx":004E
      Left            =   2640
      List            =   "frm_sort_results.frx":0064
      TabIndex        =   2
      Top             =   1320
      Width           =   1335
   End
   Begin VB.ListBox lst_first_sort
      Height          =   1230
      ItemData        =   "frm_sort_results.frx":009C
      Left            =   600
      List            =   "frm_sort_results.frx":00B2
      TabIndex        =   0
      Top             =   1320
      Width           =   1335
   End
   Begin VB.Label Label3
```

```
      Caption         =   "Third Sort Variable"
      Height          =   375
      Left            =   4800
      TabIndex        =   5
      Top             =   720
      Width           =   1335
   End
   Begin VB.Label Label2
      Caption         =   "Second Sort Variable"
      Height          =   375
      Left            =   2520
      TabIndex        =   3
      Top             =   720
      Width           =   1815
   End
   Begin VB.Label Label1
      Caption         =   "First Sort Variable"
      Height          =   375
      Left            =   480
      TabIndex        =   1
      Top             =   720
      Width           =   1335
   End
End
Attribute VB_Name = "frm_sort_results"
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = False
Attribute VB_PredeclaredId = True
Attribute VB_Exposed = False

Private Sub but_cancel_Click()
Me.Hide
Unload Me

End Sub

Private Sub but_Sort_Click()
Dim max_val As Single, max_dig As Integer, fformat As String
max_val = -99999999
Me.Hide
'setup_data
With frm_main.spr_result
' if using columns 1 or 4, format the data
If Me.lst_first_sort.ListIndex = 0 Or Me.lst_second_sort.ListIndex = 0
_
Or Me.lst_third_sort.ListIndex = 0 Then col_format (1)
If Me.lst_first_sort.ListIndex = 3 Or Me.lst_second_sort.ListIndex = 3
_
Or Me.lst_third_sort.ListIndex = 3 Then col_format (4)
If Me.lst_first_sort.ListIndex = 4 Or Me.lst_second_sort.ListIndex = 4
_
Or Me.lst_third_sort.ListIndex = 4 Then col_format (8)
If Me.lst_first_sort.ListIndex = 5 Or Me.lst_second_sort.ListIndex = 5
_
Or Me.lst_third_sort.ListIndex = 5 Then col_format (9)
Dim key1 As Integer, key2 As Integer, key3 As Integer
    If Me.lst_first_sort.ListIndex = -1 Then
```

```
MsgBox ("Please select one or more sort keys")
Exit Sub
End If
key1 = Me.lst_first_sort.ListIndex + 1
If key1 = 5 Then key1 = 8
If key1 = 6 Then key1 = 9
key2 = Me.lst_second_sort.ListIndex + 1
If key2 = 5 Then key2 = 8
If key2 = 6 Then key2 = 9
key3 = Me.lst_third_sort.ListIndex + 1
If key3 = 5 Then key3 = 8
If key3 = 6 Then key1 = 9

.col = 1
.Col2 = 9
.row = 1
.Row2 = run_number
.SortKey(1) = key1
.SortKeyOrder(1) = 1
If key2 > 0 Then
.SortKey(2) = key2
.SortKeyOrder(2) = 1
End If
If key3 > 0 Then
.SortKey(3) = key3
.SortKeyOrder(3) = 1
End If
' we need to format columns 1 and 4
If key1 = 2 Or key1 = 3 Then .SortKeyOrder(1) = 2
If key2 = 2 Or key2 = 3 Then .SortKeyOrder(2) = 2
If key3 = 2 Or key3 = 3 Then .SortKeyOrder(3) = 2
.SortBy = 0
.Action = 25
If Me.lst_first_sort.ListIndex = 0 Or Me.lst_second_sort.ListIndex = 0
_
Or Me.lst_third_sort.ListIndex = 0 Then col_unformat (1)
If Me.lst_first_sort.ListIndex = 3 Or Me.lst_second_sort.ListIndex = 3
_
Or Me.lst_third_sort.ListIndex = 3 Then col_unformat (4)
If Me.lst_first_sort.ListIndex = 4 Or Me.lst_second_sort.ListIndex = 4
_
Or Me.lst_third_sort.ListIndex = 4 Then col_unformat (8)
If Me.lst_first_sort.ListIndex = 5 Or Me.lst_second_sort.ListIndex = 5
_
Or Me.lst_third_sort.ListIndex = 5 Then col_unformat (9)
End With
Unload Me
End Sub
Sub col_format(coll As Integer)
Dim this_row As Integer, max_val As Single, max_dig As Integer
max_val = -99999999
With frm_main.spr_result
 .col = coll
For this_row = 1 To run_number
 .row = this_row
 If Val(.text) > max_val Then max_val = Val(.text)
Next this_row
```

```
 If max_val <> 0 Then
max_dig = Log(max_val) / Log(10)
Else
MsgBox ("results cannot be sorted")
Exit Sub
End If
fformat = String(max_dig + 1, "0") & "." & String(8 - max_dig, "#")
' and format all the data
For this_row = 1 To run_number
.row = this_row
.text = Format(Val(.text), fformat)
Next this_row
End With
End Sub
Sub setup_data()
Dim this_row As Integer
With frm_main.spr_result
frm_main.SSTab1.Tab = 2
For this_row = 1 To 40
.row = this_row
.col = 1
.text = (Rnd() * 4) ^ 3
.col = 2
.text = this_row Mod 2
.col = 3
.text = this_row Mod 3
.col = 4
.text = (Rnd() * 4) ^ 3
.col = 8
.text = this_row - (this_row Mod 10)
.col = 9
.text = this_row Mod 10
Next this_row
End With
End Sub
Sub col_unformat(col As Integer)
With frm_main.spr_result
.col = col
For this_row = 1 To run_number
.row = this_row
.text = Val(.text)
Next this_row
End With
End Sub
Sub new_sort()
Dim max_val As Single, max_dig As Integer, fformat As String
max_val = -9999999
Me.Hide
With frm_main.spr_result
For this_row = 1 To 1000
.col = 1
 .row = this_row
 .text = (Rnd() * 10) ^ 3
 If Val(.text) > max_val Then max_val = Val(.text)
.col = 2
  .text = this_row Mod 2
Next this_row
```

File frm_text.frm

```
VERSION 5.00
Object = "{F9043C88-F6F2-101A-A3C9-08002B2F49FB}#1.1#0"; "Comdlg32.ocx"
Begin VB.Form frm_text
   Caption         =   "Form1"
   ClientHeight    =   10830
   ClientLeft      =   300
   ClientTop       =   630
   ClientWidth     =   15075
   LinkTopic       =   "Form1"
   ScaleHeight     =   10830
   ScaleWidth      =   15075
   Begin MSComDlg.CommonDialog CommonDialog1
      Left            =   2520
      Top             =   1680
      _ExtentX        =   847
      _ExtentY        =   847
      _Version        =   327680
   End
   Begin VB.TextBox txt_text
      BeginProperty Font
         Name            =   "Courier"
         Size            =   9.75
         Charset         =   0
         Weight          =   400
         Underline       =   0   'False
         Italic          =   0   'False
         Strikethrough   =   0   'False
      EndProperty
      Height          =   10815
      Left            =   0
      Locked          =   -1  'True
      MultiLine       =   -1  'True
      ScrollBars      =   3  'Both
      TabIndex        =   0
      Top             =   0
      Width           =   15000
   End
   Begin VB.Menu file
      Caption         =   "File"
      Begin VB.Menu copy
         Caption         =   "Copy"
      End
      Begin VB.Menu exit
         Caption         =   "Exit"
      End
   End
   Begin VB.Menu edit
      Caption         =   "Edit"
      Begin VB.Menu font
         Caption         =   "Font"
      End
   End
End
Attribute VB_Name = "frm_text"
Attribute VB_GlobalNameSpace = False
```

```
Attribute VB_Creatable = False
Attribute VB_PredeclaredId = True
Attribute VB_Exposed = False
Private Sub Command1_Click()
Me.Hide
frm_main.Show
End Sub

Private Sub copy_Click()
Dim copy_string As String
If txt_text.SelStart = 0 Then
MsgBox "No text selected"
Exit Sub
End If
copy_string = Mid(txt_text, txt_text.SelStart, txt_text.SelLength)
Clipboard.SetText copy_string     ' Put text on Clipboard.
End Sub

Private Sub exit_Click()
Me.Hide
frm_main.Show
End Sub

Private Sub font_Click()
CommonDialog1.FontBold = txt_text.FontBold
CommonDialog1.FontItalic = txt_text.FontItalic
CommonDialog1.FontName = txt_text.FontName
CommonDialog1.FontSize = txt_text.FontSize
   CommonDialog1.CancelError = True
    On Error GoTo ErrHandler
    ' Set the Flags property
    CommonDialog1.Flags = cdlCFEffects Or cdlCFBoth
    ' Display the Font dialog box
    CommonDialog1.ShowFont
    txt_text.Font.Name = CommonDialog1.FontName
    txt_text.Font.Size = CommonDialog1.FontSize
    txt_text.Font.Bold = CommonDialog1.FontBold
    txt_text.Font.Italic = CommonDialog1.FontItalic
    txt_text.Font.Underline = CommonDialog1.FontUnderline
    txt_text.FontStrikethru = CommonDialog1.FontStrikethru
    txt_text.ForeColor = CommonDialog1.Color
    Exit Sub
ErrHandler:
    ' User pressed the Cancel button
    Exit Sub

'CommonDialog1.FontBold = txt_text.FontBold
'CommonDialog1.FontItalic = txt_text.FontItalic
'CommonDialog1.FontName = txt_text.FontName
'CommonDialog1.FontSize = txt_text.FontSize
'CommonDialog1.ShowFont
'txt_text.FontName = CommonDialog1.FontName
'txt_text.FontBold = CommonDialog1.FontBold
'txt_text.FontItalic = CommonDialog1.FontItalic
'txt_text.FontSize = CommonDialog1.FontSize
End Sub
```

```
Private Sub Form_Terminate()
Me.Hide
frm_main.Show
End Sub

Private Sub Form_Unload(Cancel As Integer)
Me.Hide
frm_main.Show
End Sub
```

File frm_tokens.frm

```
VERSION 5.00
Begin VB.Form frm_tokens
   Caption         =   "Tokens"
   ClientHeight    =   6360
   ClientLeft      =   1620
   ClientTop       =   1890
   ClientWidth     =   11910
   LinkTopic       =   "Form1"
   ScaleHeight     =   6360
   ScaleWidth      =   11910
   Begin VB.ListBox lst_token_group
      Height          =   3375
      Left            =   9240
      TabIndex        =   10
      Top             =   960
      Width           =   2175
   End
   Begin VB.CommandButton but_remove_group
      Caption         =   "Remove Token Group"
      Height          =   495
      Left            =   10200
      TabIndex        =   9
      Top             =   4680
      Width           =   1575
   End
   Begin VB.CommandButton but_new_group
      Caption         =   "New Token Group"
      Height          =   495
      Left            =   8640
      TabIndex        =   8
      Top             =   4680
      Width           =   1455
   End
   Begin VB.CommandButton but_new_set
      Caption         =   "New Token set"
      Height          =   495
      Left            =   4200
      TabIndex        =   5
      Top             =   4680
      Width           =   1455
   End
   Begin VB.CommandButton but_remove_set
      Caption         =   "Remove Token set"
      Height          =   495
      Left            =   6240
      TabIndex        =   3
      Top             =   4680
      Width           =   1575
   End
   Begin VB.ListBox lst_token_sets
      Height          =   3375
      Left            =   3240
      TabIndex        =   2
      Top             =   960
      Width           =   5295
```

```
         Underline        =   0   'False
         Italic           =   0   'False
         Strikethrough    =   0   'False
      EndProperty
      Height           =   375
      Left             =   1080
      TabIndex         =   6
      Top              =   0
      Width            =   1335
   End
   Begin VB.Label lbl_token_name
      Height           =   375
      Left             =   4560
      TabIndex         =   4
      Top              =   480
      Width            =   1935
   End
End
Attribute VB_Name = "frm_tokens"
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = False
Attribute VB_PredeclaredId = True
Attribute VB_Exposed = False
Dim curr_group As String
Dim curr_position As Integer ' which token group
Dim curr_token As Integer ' which token
Dim curr_token_set As Integer ' which token set
Private Sub but_done_Click()
Me.Hide
frm_main.Show
End Sub
'
Private Sub but_new_group_Click()
Me.Hide
frm_new_group.txt_n_tokens = ""
frm_new_group.txt_n_tokens = "1"
frm_new_group.Show modal:=1, ownerform:=Me
' n_tokens is set to -999 by cancel
If frm_new_group.txt_n_tokens = "-999" And frm_new_group.txt_stem = "-
999" Then
Exit Sub
End If
n_token_groups = n_token_groups + 1
Dim curr_stem As String
curr_stem = frm_new_group.txt_stem
token_collection(n_token_groups).stem = curr_stem
token_collection(n_token_groups).n_tokens = frm_new_group.txt_n_tokens
lst_token_group.AddItem curr_stem
Me.lbl_token_name = curr_stem
curr_group = curr_stem
curr_position = n_token_groups
lst_token_group.ListIndex = curr_position - 1
'token_collection(curr_position).n_token_sets = 0
token_collection(curr_position).get_token_set lst_token_sets
lst_tokens.clear
End Sub
```

```
Private Sub but_new_set_Click()
If curr_position > 0 Then
token_collection(curr_position).add_token_set lst_token_sets
curr_token_set = lst_token_sets.ListCount
token_collection(curr_position).get_tokens lst_tokens, curr_token_set
lst_token_sets.ListIndex = lst_token_sets.ListCount - 1
End If
End Sub

Private Sub but_remove_group_Click()
Dim this_group As Integer
If lst_token_group.ListIndex > -1 Then
n_token_groups = n_token_groups - 1
 For this_group = lst_token_group.ListIndex + 1 To n_groups - 1
  Set token_collection(this_group) = token_collection(this_group + 1)
 Next this_group
 Set token_collection(n_groups) = Nothing
 lst_token_group.RemoveItem (lst_token_group.ListIndex)
 token_collection(curr_position).get_token_set lst_token_sets
 If lst_token_group.ListCount > 0 Then
 lst_token_group.ListIndex = 0
 Else
 lst_token_group.ListIndex = -1
 lst_token_sets.clear
 End If
  curr_position = 1
  token_collection(curr_position).get_token_set lst_token_sets
 If lst_token_sets.ListCount > 0 Then
 lst_token_sets.ListIndex = 0
 Else
 lst_token_sets.ListIndex = -1
 lst_tokens.clear
 End If
End If
End Sub

Private Sub but_remove_set_Click()
If lst_token_sets.ListIndex > -1 Then
token_collection(curr_position).remove_token_set
(lst_token_sets.ListIndex + 1)
token_collection(curr_position).get_token_set lst_token_sets
curr_position = lst_token_group.ListIndex + 1
token_collection(curr_position).get_token_set lst_token_sets
If lst_token_sets.ListCount > 0 Then
lst_token_sets.ListIndex = 0
Else
lst_token_sets.ListIndex = -1
lst_tokens.clear
End If
curr_token_set = lst_token_sets.ListIndex + 1
token_collection(curr_position).get_tokens lst_tokens, curr_token_set
End If
End Sub
```

```
Private Sub Form_Load()
curr_position = 1
curr_token_set = 1
curr_token = 1
End Sub

Private Sub lst_token_group_Click()
curr_group = lst_token_group.list(lst_token_group.ListIndex)
curr_position = lst_token_group.ListIndex + 1
token_collection(curr_position).get_token_set lst_token_sets

curr_token_set = 1
If lst_token_sets.ListCount > 0 Then
lst_token_sets.ListIndex = curr_token_set - 1
End If
token_collection(curr_position).get_tokens lst_tokens, curr_token_set
End Sub

Private Sub lst_token_sets_Click()
curr_token_set = lst_token_sets.ListIndex + 1
token_collection(curr_position).get_tokens lst_tokens, curr_token_set
End Sub

Private Sub lst_tokens_dblClick()
curr_token = lst_tokens.ListIndex + 1
frm_edit_token.txt_token =
token_collection(curr_position).get_token_with_lines(curr_token_set,
curr_token)
Me.Hide

frm_edit_token.Show modal:=1, ownerform:=Me
' first change the token in the token set
token_collection(curr_position).set_token curr_token_set, curr_token,
frm_edit_token.txt_token
' update the tokenn list
token_collection(curr_position).get_tokens lst_tokens, curr_token_set
' update token set list
token_collection(curr_position).get_token_set lst_token_sets
End Sub
```